S^2-PINN: Stochastic Separable Physics-Informed Neural Networks

Zhendong Li
Industrial and Systems Engineering Dept.
Lehigh University
Bethlehem, PA 18015, USA
zhl923@lehigh.edu

Akwum Onwunta
Industrial and Systems Engineering Dept.
Lehigh University
Bethlehem, PA 18015, USA
ako221@lehigh.edu

Abstract

Uncertainty quantification (UQ) for random partial differential equations (PDEs) is ubiquitous in computational science and engineering. However, classical spectral solvers for this class of problems face the curse of dimensionality, and existing neural solvers often ignore the stochastic structure that makes moments and calibration tractable. We introduce a stochastic separable physics-informed neural network, dubbed S^2-PINN, that represents the solution $u(t, \mathbf{x}, \mathbf{Z})$ of a random PDE with a learnable Gaussian spatial dictionary, Fourier temporal features, and a generalized polynomial chaos (gPC) stochastic basis, coupled by a low-rank Canonical Polyadic (CP) tensor decomposition core. The method is trained with a hybrid strong-form and gPC-projected residual loss. Our theoretical analysis establishes that the separable class is dense in L^2 under mild conditions, and the projected residual corresponds exactly to a stochastic Galerkin constraint. Furthermore, we show that mini-batch projection coefficients are logarithmically dependent on the number of gPC modes, and that the orthogonality penalty controls the conditioning of the learned spatial dictionary. Using four manufactured random PDE benchmarks, we show that S^2-PINN outperforms nine baselines in terms of mean and variance accuracy, as well as calibration, while using significantly fewer parameters. Further evaluations on non-manufactured Poisson and Darcy problems, a stochastic Navier–Stokes problem, a diffusion scaling study of higher random dimensions, and two stochastic inverse problems reveal the generalization capabilities of the proposed structure. Together, these results support stochastic separability as an effective design principle for physics-informed neural UQ. The code for the experiments can be found in `https://github.com/DMax1314/s2pinn`.

1 Introduction

Uncertainty quantification (UQ) for partial differential equations (PDEs) is essential when coefficients, boundary conditions, or forcing terms depend on random variables $\mathbf{Z} \in \mathbb{R}^R$. In this setting, the solution $u(t, \mathbf{x}, \mathbf{Z})$ is a random field, and the primary targets are often its moments and calibration properties. Classical stochastic solvers such as Monte Carlo Ghanem and Spanos [1991], stochastic Galerkin Xiu and Karniadakis [2002], and stochastic collocation Babuška et al. [2007] are reliable but costly because they require many repeated solves or large coupled systems.

Physics-informed neural networks (PINNs) Raissi et al. [2019], Karniadakis et al. [2021] avoid labeled data by enforcing PDE residual constraints, but standard PINNs typically encode $\mathbf{Z}$ as generic inputs and rely on dense sampling to learn stochastic structure implicitly. Operator-learning methods such as DeepONet Lu et al. [2021] and FNO Li et al. [2020] learn parameter-to-solution maps from data, but require supervised datasets and degrade in stiff nonlinear regimes. Recent stochastic PINN and physics-informed polynomial-chaos work Zhang et al. [2019], Yang et al. [2021], Psaros et al.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

[2023], Novák et al. [2024], Shi et al. [2026] has improved neural UQ, yet current architectures do not jointly exploit separability across space, time, and randomness.

Separable PINNs (SPINN) Cho et al. [2023] showed that explicit separability can improve efficiency for deterministic PDEs. In parallel, low-rank tensor methods (CP, TT, Tucker) Kolda and Bader [2009], Oseledets [2011], Bachmayr [2023] have long addressed high-dimensional structure in PDE and optimal control solvers Nouy [2010], Chinesta et al. [2010], Bachmayr et al. [2016], Benner et al. [2015, 2016a,b], Antil et al. [2023, 2025a,b]. The missing ingredient is an integrated stochastic PINN that combines learnable spatial representations, explicit stochastic basis structure, and physics constraints enforcing both pointwise residual consistency and stochastic projection consistency.

S^2-PINN fills this gap. The model represents the solution $u(t, \mathbf{x}, \mathbf{Z})$ by a three-way separable ansatz with a learnable Gaussian spatial dictionary, Fourier temporal features, and gPC stochastic modes, coupled through a low-rank CP core. Training combines strong-form residual minimization with a gPC-projected residual term. Our theory supports four concrete claims: finite-rank separable sums are expressive in L^2, the projected residual equals a stochastic Galerkin constraint, its mini-batch estimator has uniform concentration scaling as $O(\log G)$, where G is the number of gPC modes, and the spatial orthogonality penalty controls feature conditioning. We evaluate both forward stochastic prediction and inverse stochastic parameter identification within the same physics-informed framework.

Contributions. In this work, we make the following three main contributions.

(a) We introduce a stochastic separable PINN that couples a learnable spatial dictionary with M atoms, M_t temporal Fourier features, and G gPC stochastic modes via a rank-q CP factorization, reducing coupling complexity from dense $O(MM_tG)$ to $O((M+M_t+G)q)$. The hybrid strong-form and gPC-projected residual loss enforces local PDE consistency and stochastic Galerkin consistency.

(b) We prove expressivity of the separable ansatz, projection–Galerkin equivalence, mini-batch concentration for stochastic projection coefficients, orthogonality-conditioning control, CP complexity, and gauge invariance. These results justify the representation and explain why projection is useful when the solution is well aligned with the chosen stochastic basis.

(c) Among manufactured and non-manufactured benchmarks, S^2-PINN improves moment accuracy and calibration. Additional scaling and inverse-problem studies extend the evaluation beyond standard forward prediction, covering higher-dimensional random inputs and joint recovery of solution fields and random-field parameters.

Our central claim is that stochastic separability provides a compact and well-aligned inductive bias for neural UQ of random PDEs, with its strongest benefits in regimes where the solution has low-rank space–time–random structure and a useful low-order stochastic expansion.

2 Related Work

Physics-Informed and Operator Learning for UQ. Physics-informed neural networks typically enforce PDEs via soft residual constraints Raissi et al. [2019], with recent efforts targeting Uncertainty Quantification (UQ) via Bayesian frameworks Yang et al. [2021], auxiliary networks Zhang et al. [2019], and polynomial chaos expansions Novák et al. [2024], Shi et al. [2026]. Parallel to PINNs, neural operators like DeepONet Lu et al. [2021] and FNO Li et al. [2020] learn parameter-to-solution maps entirely from data. Very recently, Neural Chaos Bahmani et al. [2025] proposed replacing fixed polynomial bases with sequentially learned networks to break the independence assumption. However, a common limitation spans these methods: they either inherently mandate extensive labeled training data (data-driven operators), fail to exploit multidimensional structural separation, or rely on dense representations that scale poorly with stochastic dimensions. S^2-PINN addresses this by formulating a strictly physics-informed, uncoupled architecture that mathematically isolates spatial derivative costs from the stochastic basis size. Physics-informed polynomial chaos expansions (PC^2) Novák et al. [2024] are particularly relevant to our setting because they impose PDE constraints directly on a PCE surrogate via KKT-style constrained regression. We therefore include a UQpy-based PC^2 comparison as an external polynomial-chaos baseline: it tests whether a fixed global polynomial surrogate with physics constraints can match the accuracy and calibration gains obtained from our learnable space–time representation and low-rank stochastic coupling.

Low-Rank Tensor Methods in Neural Solvers. Classical low-rank tensor decompositions (CP, TT, Tucker) have long accelerated stochastic Galerkin solvers Doostan et al. [2013], Bigoni et al. [2016]. In the deep learning regime, recent state-of-the-art neural-tensor hybrids—such as TT-PINN Liu et al. [2022], functional tensor decompositions Vemuri et al. [2024], and INN-TD Guo et al. [2025]—target purely *deterministic* spatial/parametric coordinates. They primarily compress neural weight matrices for edge deployment or factorize spatial axes without incorporating a statistical structural prior. Because none embed a stochastic polynomial basis or a UQ-specific Galerkin projection, they cannot address robust uncertainty quantification. S^2-PINN fundamentally diverges by fusing a learnable spatial dictionary with explicitly separated gPC stochastic modes, governed entirely by a hybrid strong-form and projection constraint.

Separable PINNs for High-Dimensional PDEs. SPINN Cho et al. [2023] first demonstrated that explicit dimensional separability, coupled with a coordinate warp, drastically accelerates deterministic PDE training. While it has been extended to quantum and elastic systems Es'kin et al. [2024], Zanotta et al. [2025], seamlessly scaling SPINN to stochastic parameters is highly non-trivial. Naively introducing random variables into SPINN yields a dense, non-factorized coefficient tensor that grows multiplicatively with the spatial, temporal, and stochastic resolutions. S^2-PINN mitigates this specific bottleneck by eschewing the dense multi-way block for a low-rank CP coupling kernel, tightly integrating optimal stochastic bases (gPC) with spatio-temporal atoms, a structural synthesis currently absent in the SPINN-related literature. To isolate the effect of this design, we additionally compare against structurally closer baselines: SPINN extended with stochastic coordinates (SPINN+Z) Cho et al. [2023], functional CP-PINNs Vemuri et al. [2024], and Neural Chaos Bahmani et al. [2025].

3 Method

S^2-PINN represents random PDE solutions with an explicitly separated space–time–random structure. The method assumes that the uncertainty is encoded by a finite random vector $\mathbf{Z}$ with known marginals, which allows us to choose an orthonormal gPC basis in the stochastic coordinates. The architecture then couples spatial, temporal, and stochastic factors through a low-rank CP kernel, while the loss combines pointwise physics constraints with a stochastic projection constraint. Appendix B provides PDE-agnostic support for the main design choices: expressivity of separable sums, projection–Galerkin equivalence, mini-batch concentration, orthogonality penalty conditioning, CP complexity, and gauge invariance.

3.1 Three-way Separable Representation

We approximate the solution by a rank-q CP expansion,

$$u(t, \mathbf{x}, \mathbf{Z}) \approx \sum_{r=1}^{q} w_r \phi^{(r)}(\mathbf{x}) \tau^{(r)}(t) \psi^{(r)}(\mathbf{Z}), \tag{1}$$

where each factor is a linear combination of basis functions (atoms) specific to its coordinate,

$$\phi^{(r)}(\mathbf{x}) = \sum_{i=1}^{M} A_{ir} \Phi_i(\mathbf{x}), \quad \tau^{(r)}(t) = \sum_{k=1}^{M_t} B_{kr} T_k(t), \quad \psi^{(r)}(\mathbf{Z}) = \sum_{g=1}^{G} D_{gr} \Psi_g(\mathbf{Z}). \tag{2}$$

Here $A \in \mathbb{R}^{M \times q}$, $B \in \mathbb{R}^{M_t \times q}$, $D \in \mathbb{R}^{G \times q}$, and $w \in \mathbb{R}^q$ are learnable CP parameters. Spatial atoms are Gaussian dictionary elements with an optional residual warp,

The three feature families are as follows. Spatial atoms are Gaussian dictionary elements with an optional residual warp,

$$\Phi_i(\mathbf{x}) = \exp\Big(- \tfrac{\|\mathbf{W}(\mathbf{x}) - \mathbf{c}_i\|_2^2}{2\sigma_i^2} \Big), \qquad \mathbf{W}(\mathbf{x}) = \mathbf{x} + \mathbf{w}_\theta(\mathbf{x}). \tag{3}$$

Temporal atoms are DC and sine/cosine features with learnable frequencies $\omega_k = \exp(\theta_k)$, and stochastic atoms are tensorized gPC bases (chosen by the base distribution) on $\mathcal{J}_{R,p}$. Given feature blocks $\mathbf{F}_{\text{space}} \in \mathbb{R}^{B \times M}$, $\mathbf{F}_{\text{time}} \in \mathbb{R}^{B \times M_t}$, and $\mathbf{F}_{\text{stoc}} \in \mathbb{R}^{B \times G}$, the batched forward map is

$$u = \big((\mathbf{F}_{\text{space}} A) \odot (\mathbf{F}_{\text{time}} B) \odot (\mathbf{F}_{\text{stoc}} D) \big) w, \tag{4}$$

where $\odot$ denotes a row-wise Hadamard product. Arithmetic scales as $O(Bq(M + M_t + G))$ instead of $O(BMM_tG)$ (Theorem B.9). Relative to SPINN, this design adds explicit stochastic modes, learns a Gaussian spatial dictionary, and uses a three-way CP coupling that keeps the stochastic basis separated from spatial derivative costs. Figure 1 illustrates the architecture.

Theorem 3.1. *Under the density assumptions on $\{\Phi_i\}$, $\{T_k\}$, and $\{\Psi_g\}$ stated in Appendix B, the union $\bigcup_{q\geq 1} \mathcal{F}_q$ of rank-q separable functions is dense in $L^2(\mathcal{T} \times \Omega \times \mathbf{Z})$; see Theorem B.2.*

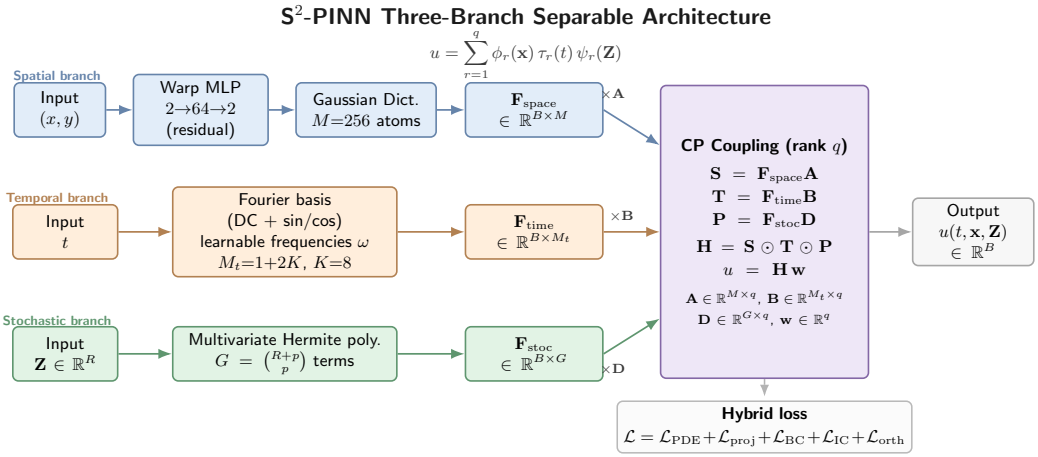

Figure 1: S^2-PINN architecture. Three specialized branches—spatial (Gaussian dictionary with warp), temporal (learnable Fourier), stochastic (Hermite gPC)—are coupled via a rank-q CP core using Hadamard products. The hybrid loss combines PDE residual, gPC projection, boundary/initial conditions, and orthogonality regularization.

3.2 Hybrid Physics Loss and Regularization

Let $r_\theta = \mathcal{N}[u_\theta; \vartheta](t, \mathbf{x}, \mathbf{Z})$ be the strong-form residual, where ϑ parameterizes the differential operator $\mathcal{N}$ and u_θ is the network output. We minimize the objective

$$\mathcal{L} = w_{\text{pde}}\mathbb{E}[r_\theta^2] + w_{\text{proj}} \sum_{g=1}^{G} \Big(\mathbb{E}[r\Psi_g(\mathbf{Z})]\Big)^2 + w_{\text{bc}}\mathbb{E}_{\partial\Omega}[(u - u_\partial)^2] + w_{\text{ic}}\mathbb{E}[(u - u_0)^2|_{t=0}] + w_{\text{orth}}\mathcal{R}_{\text{orth}}. \tag{5}$$

Here u_∂ and u_0 denote boundary and initial conditions. The orthogonal regularizer combines a spatial feature Gram penalty

$$\mathcal{R}_{\text{feat}} = \|\widehat{G}_\Phi - I\|_F^2, \qquad \widehat{G}_\Phi = \tfrac{1}{B}\mathbf{F}_{\text{space}}^\top \mathbf{F}_{\text{space}}, \tag{6}$$

and CP-factor orthogonality,

$$\mathcal{R}_{\text{fac}} = \sum_{M\in\{A,B,D\}} \Big\| \tfrac{1}{\|M\|_{\text{col}}^2} M^\top M - I \Big\|_F^2, \tag{7}$$

with $\mathcal{R}_{\text{orth}} = \mathcal{R}_{\text{feat}} + \lambda_{\text{fac}}\mathcal{R}_{\text{fac}}$. Note that the CP format is defined by the forward map (4) itself, which is an explicit sum of rank-one products of spatial, temporal, and stochastic factors for any value of λ_{fac}, including zero; the factor penalty (default λ_{fac}=0.1) only encourages well-conditioned factors and neither defines the CP form nor enforces CP uniqueness. Theorem B.3 shows that near-identity spatial Gram matrices control the condition number of the feature matrix.Regularization incorporates a spatial Gram penalty, which encourages feature near-orthogonality. Specifically, Theorem B.7 shows that $\|\widehat{G}_\Phi - I\|_2 \leq \delta$ ensures $\kappa(F_{\text{space}}) \leq \sqrt{(1+\delta)/(1-\delta)}$, with Theorem B.8 providing the corresponding sample complexity.

The projected residual term is theoretically Galerkin-consistent and statistically stable, but it is not uniformly beneficial in every stochastic regime. Theorem B.3 identifies $\sum_g (\mathbb{E}[r_\theta \Psi_g])^2$ with $\|\mathsf{P}_{V_p} r_\theta\|^2_{L^2(\mathbf{Z})}$, the squared residual projection onto the gPC subspace. The mini-batch coefficient

estimator is unbiased with $O(1/B)$ variance (Theorem B.4), and Theorem B.5 gives uniform ϵ-accuracy across all G modes with batch size scaling as $O(\log G)$. Empirically, this projection acts as a useful spectral regularizer when the solution is well approximated by the chosen polynomial space, but it can introduce bias when important stochastic energy lies outside that space. In practice, the projection is computed pointwise in space and time: for each sampled $(t, \mathbf{x})$ we draw several stochastic samples $\{\mathbf{Z}_b\}$, estimate $\mathbb{E}_{\mathbf{Z}}[r_\theta(t, \mathbf{x}, \mathbf{Z})\Psi_g(\mathbf{Z})]$ from them, and then average the squared coefficients over the space–time collocation points, rather than taking one joint average over random $(t, \mathbf{x}, \mathbf{Z})$ tuples.

3.3 Training

At each step we sample PDE, boundary, initial, and Gram mini-batches and optimize Eq. (5) with Adam. Time derivatives are analytic from Fourier features, while spatial derivatives, including Δu, use autodiff through the warped Gaussian dictionary. We periodically resample collocation points and apply column-wise normalization to CP factors (A, B, D) after updates to fix the continuous gauge symmetry of the CP format (Theorem B.10), which also improves the Lipschitz conditioning of the forward map. The column normalization leaves the forward map unchanged (Theorem B.10) but rescales Adam's moment buffers; at the default learning rate $\eta = 10^{-3}$, we observed no instability from this engineering choice. Algorithm 1 and default hyperparameters are reported in Appendix E.

4 Experiments

4.1 Benchmark Problems

We evaluate S^2-PINN on four controlled random PDEs: diffusion, Allen–Cahn, Burgers, and Darcy. The first three are time-dependent problems that share the same manufactured target and differ in the physical operator, so their performance gaps isolate the effect of linear diffusion, cubic reaction, and nonlinear advection. Darcy is steady-state and uses exponential $\mathbf{Z}$-dependence in the coefficient, making it a deliberately harder case for a low-order polynomial stochastic basis.

Unless otherwise stated, all experiments use R=5 random variables, total gPC degree p=3 (G=$\binom{8}{3}$=56 modes), M=256 spatial atoms, M_t=17 temporal atoms, and CP rank q=16. We keep this single default across the four PDEs, the scaling sweep (Section G), and the collocation-density sweep (Section H) to avoid per-problem tuning. Manufactured solutions enable ground-truth evaluation of both mean and variance; full formulas are given in Section C.

Baselines and fairness. We compare against nine baselines: a standard MLP PINN, the same PINN with the gPC projection loss, PI-DeepONet Goswami et al. [2023], MC-Dropout PINN, a K=5 deep ensemble Lakshminarayanan et al. [2017], stochastic collocation (SC) with n_q=2 Gauss–Hermite nodes per dimension (2^5=32 node-PINNs), and three structurally closer methods: SPINN with the random variables appended as additional separable coordinates (SPINN+Z) Cho et al. [2023], functional CP-PINN Vemuri et al. [2024], and Neural Chaos Bahmani et al. [2025]. Neural Chaos is supervised by reference solutions and therefore serves as a data-driven oracle rather than a purely physics-informed competitor. All neural methods use the same optimizer family, collocation strategy, and training budget unless stated otherwise. The vanilla PINN has about 28$\times$ more parameters than S^2-PINN (roughly 200K versus 7K), so parameter count does not favor the proposed architecture. Additional architecture and optimization details are in Appendix D.

4.2 Metrics

We report two moment-accuracy metrics, evaluated on a held-out stochastic test set:

- **Mean relative** L^2 : $\|\mathbb{E}_{\mathbf{Z}}[u] - \mathbb{E}_{\mathbf{Z}}[u^*]\|_{L^2(\Omega)} / \|\mathbb{E}_{\mathbf{Z}}[u^*]\|_{L^2(\Omega)}$
- **Variance relative** L^2 : $\|\mathrm{Var}_{\mathbf{Z}}[u] - \mathrm{Var}_{\mathbf{Z}}[u^*]\|_{L^2(\Omega)} / \|\mathrm{Var}_{\mathbf{Z}}[u^*]\|_{L^2(\Omega)}$

Moments are estimated by Monte Carlo over 1024 i.i.d. samples $\mathbf{Z} \sim \mathcal{N}(0, I_R)$ at B=2048 uniformly sampled probe points $(t, \mathbf{x}) \in \mathcal{T} \times \Omega$. All results are averaged over 5 random seeds (42, 123, 456, 789, 1024), and all results reported as mean $\pm$ std. The lowest mean per column is bolded and when the runner-up lies within $\pm 1\sigma$ of the best, both entries are bolded to indicate a statistical tie.

Calibration protocol. To assess whether the predicted uncertainty faithfully reflects the true variability of the solution under random inputs, we construct predictive intervals and evaluate their empirical coverage. At each probe $(t_j, \mathbf{x}_j)$, we draw N_Z=1,024 i.i.d. samples $\{\mathbf{Z}^{(i)}\}_{i=1}^{N_Z}$ from $\mathcal{N}(0, I_R)$ and form the nominal $(1-\alpha)$ predictive interval $[\hat{Q}_{\alpha/2},\ \hat{Q}_{1-\alpha/2}]$, where $\hat{Q}_\tau$ denotes the τ-th empirical quantile of $\{u_\theta(t_j, \mathbf{x}_j, \mathbf{Z}^{(i)})\}$. Coverage is then evaluated on a separate hold-out set of N'_Z=1,024 samples $\{\mathbf{Z}'^{(k)}\}_{k=1}^{N'_Z}$: empirical coverage at nominal level $(1-\alpha)$ is the fraction of (j,k) pairs for which $u^*(t_j, \mathbf{x}_j, \mathbf{Z}'^{(k)}) \in [\hat{Q}^{(j)}_{\alpha/2},\ \hat{Q}^{(j)}_{1-\alpha/2}]$. We report empirical coverage at four nominal levels (50%, 80%, 90%, 95%), sharpness (mean interval width at 90%), and mean calibration error $\frac{1}{4}\sum_\ell |(1-\alpha_\ell) - \widehat{\mathrm{Cov}}_\ell|$. Results are in Table 3.

4.3 Main Results

Tables 1–2 summarize the comparison. These estimates provide tighter uncertainty quantification of performance trends on these four controlled benchmarks.

Table 1: Mean solution accuracy.

Method	Diffusion	Allen–Cahn	Burgers	Darcy
S^2-PINN (proj)	$\mathbf{(3.71 \pm 1.01) \times 10^{-3}}$	$\mathbf{(4.40 \pm 1.32) \times 10^{-3}}$	$\mathbf{(3.44 \pm 1.47) \times 10^{-3}}$	$\mathbf{(1.05 \pm 0.30) \times 10^{-3}}$
S^2-PINN (no-proj)	$(4.85 \pm 2.80) \times 10^{-3}$	$(4.51 \pm 4.15) \times 10^{-3}$	$\mathbf{(3.86 \pm 1.84) \times 10^{-3}}$	$\mathbf{(9.51 \pm 2.13) \times 10^{-4}}$
Vanilla PINN	$(1.41 \pm 0.42) \times 10^{-2}$	$(1.81 \pm 0.68) \times 10^{-2}$	$(1.78 \pm 0.72) \times 10^{-2}$	$(1.09 \pm 0.28) \times 10^{-2}$
Vanilla PINN + proj	$(1.82 \pm 0.69) \times 10^{-2}$	$(1.49 \pm 0.40) \times 10^{-2}$	$(2.13 \pm 0.75) \times 10^{-2}$	$(1.57 \pm 0.60) \times 10^{-2}$
PI-DeepONet	$(5.07 \pm 2.17) \times 10^{-2}$	$(5.74 \pm 2.28) \times 10^{-2}$	$(4.32 \pm 1.96) \times 10^{-2}$	$(2.56 \pm 0.82) \times 10^{-2}$
MC-Dropout PINN	$(1.08 \pm 0.11) \times 10^{-1}$	$(9.99 \pm 0.77) \times 10^{-2}$	$(1.19 \pm 0.10) \times 10^{-1}$	$(7.45 \pm 0.47) \times 10^{-2}$
Deep Ensemble (K=5)	$(1.31 \pm 0.13) \times 10^{-2}$	$(1.84 \pm 0.44) \times 10^{-2}$	$(1.01 \pm 0.33) \times 10^{-2}$	$(2.05 \pm 0.45) \times 10^{-2}$
SC (n_q=2, 32 nodes)	$(3.52 \pm 0.11) \times 10^{-2}$	$(3.45 \pm 0.33) \times 10^{-2}$	$(3.57 \pm 0.10) \times 10^{-2}$	$(8.74 \pm 0.27) \times 10^{-2}$
SPINN+Z	$(1.57 \pm 5.10) \times 10^{-2}$	$(6.70 \pm 3.42) \times 10^{-2}$	$(3.59 \pm 3.43) \times 10^{-2}$	$(3.48 \pm 2.18) \times 10^{-2}$
Functional CP-PINN	$(9.37 \pm 4.22) \times 10^{-2}$	$(10.87 \pm 4.86) \times 10^{-2}$	$(10.87 \pm 4.86) \times 10^{-2}$	$(4.53 \pm 1.74) \times 10^{-2}$
Neural Chaos	$(2.31 \pm 1.35) \times 10^{-2}$	–	–	$(2.68 \pm 1.53) \times 10^{-2}$

Table 2: Variance solution accuracy.

Method	Diffusion	Allen–Cahn	Burgers	Darcy
S^2-PINN (proj)	$\mathbf{(4.40 \pm 1.52) \times 10^{-3}}$	$\mathbf{(4.07 \pm 1.98) \times 10^{-3}}$	$\mathbf{(4.65 \pm 2.00) \times 10^{-3}}$	$(5.41 \pm 0.86) \times 10^{-3}$
S^2-PINN (no-proj)	$(7.03 \pm 2.41) \times 10^{-3}$	$(7.32 \pm 7.19) \times 10^{-3}$	$\mathbf{(5.39 \pm 3.03) \times 10^{-3}}$	$\mathbf{(4.57 \pm 0.51) \times 10^{-3}}$
Vanilla PINN	$(3.75 \pm 0.80) \times 10^{-2}$	$(3.79 \pm 1.08) \times 10^{-2}$	$(5.12 \pm 1.81) \times 10^{-2}$	$(1.92 \pm 0.18) \times 10^{-2}$
Vanilla PINN + proj	$(4.75 \pm 1.30) \times 10^{-2}$	$(4.24 \pm 0.67) \times 10^{-2}$	$(4.86 \pm 0.71) \times 10^{-2}$	$(2.06 \pm 0.23) \times 10^{-2}$
PI-DeepONet	$(6.05 \pm 1.76) \times 10^{-2}$	$(6.58 \pm 2.25) \times 10^{-2}$	$(5.31 \pm 3.75) \times 10^{-2}$	$(7.87 \pm 1.98) \times 10^{-2}$
MC-Dropout PINN	$(1.54 \pm 0.25) \times 10^{-1}$	$(1.16 \pm 0.15) \times 10^{-1}$	$(1.63 \pm 0.22) \times 10^{-1}$	$(2.10 \pm 0.07) \times 10^{-1}$
Deep Ensemble (K=5)	$(4.08 \pm 2.13) \times 10^{-2}$	$(3.64 \pm 3.21) \times 10^{-2}$	$(4.22 \pm 1.89) \times 10^{-2}$	$(5.96 \pm 2.47) \times 10^{-2}$
SC (n_q=2, 32 nodes)	$(7.80 \pm 3.41) \times 10^{-2}$	$(3.45 \pm 2.27) \times 10^{-2}$	$(8.61 \pm 3.54) \times 10^{-2}$	$(7.13 \pm 3.97) \times 10^{-2}$
SPINN+Z	$(1.28 \pm 8.65) \times 10^{-2}$	$(1.40 \pm 4.56) \times 10^{-2}$	$(7.20 \pm 6.59) \times 10^{-3}$	$(2.95 \pm 1.05) \times 10^{-2}$
Functional CP-PINN	$(9.88 \pm 3.55) \times 10^{-2}$	$(9.62 \pm 5.83) \times 10^{-2}$	$(1.01 \pm 2.39) \times 10^{-1}$	$(7.21 \pm 5.18) \times 10^{-1}$
Neural Chaos	$(2.72 \pm 1.88) \times 10^{-1}$	–	–	$(6.56 \pm 8.21) \times 10^{-2}$

S^2-PINN achieves the lowest observed errors across benchmarks. Across diffusion, Allen–Cahn, and Burgers (polynomial stochastic dependence), we observe lower mean errors and lower variance errors for S^2-PINN with projection relative to the best included baseline. The no-projection variant remains competitive, which separates the architectural gain of the CP/gPC coupling from the additional regularization effect of the projected residual. Darcy is more nuanced: S^2-PINN attains the best mean accuracy, but the no-projection variant gives the best variance, already indicating that projection is not uniformly beneficial outside the polynomial regime.

Baselines. Stochastic collocation trains 32 independent node-PINNs and reaches only vanilla-PINN-level accuracy under our matched setting; it cannot share representations across random modes. The moment errors of PI-DeepONet remain about one order of magnitude above S^2-PINN. MC-dropout has substantially larger moment errors and under-covers in the calibration table. Deep ensembles improve stability over a single PINN, but use five full networks and remain less accurate on the controlled benchmarks. Among the structurally closer baselines, SPINN+Z is well calibrated but its moment errors are 1.5–33$\times$ larger than those of S^2-PINN, functional CP-PINN is less accurate and less calibrated, and even the supervised Neural Chaos oracle does not reach S^2-PINN accuracy, indicating that the gains stem from the gPC basis combined with CP coupling rather than from separability alone.

Table 3: UQ calibration: empirical coverage at nominal levels and calibration error. Calibration results for PI-DeepONet, SPINN+Z, functional CP-PINN, and Neural Chaos are in Table 17.

Method / PDE	Cov@50%	Cov@80%	Cov@90%	Cov@95%	Sharpness$_{90}$	Cal. Err.
S^2-PINN (proj)						
Diffusion	$\mathbf{0.491 \pm 0.012}$	$\mathbf{0.797 \pm 0.007}$	$\mathbf{0.897 \pm 0.004}$	$\mathbf{0.950 \pm 0.003}$	$\mathbf{0.194 \pm 0.006}$	$\mathbf{0.006 \pm 0.003}$
Allen–Cahn	$\mathbf{0.493 \pm 0.013}$	$\mathbf{0.800 \pm 0.008}$	$\mathbf{0.900 \pm 0.005}$	$\mathbf{0.952 \pm 0.004}$	$\mathbf{0.194 \pm 0.006}$	$\mathbf{0.006 \pm 0.001}$
Burgers	$\mathbf{0.493 \pm 0.012}$	$\mathbf{0.799 \pm 0.007}$	$\mathbf{0.899 \pm 0.004}$	$\mathbf{0.951 \pm 0.002}$	$\mathbf{0.194 \pm 0.006}$	$\mathbf{0.006 \pm 0.002}$
Darcy	$\mathbf{0.497 \pm 0.007}$	$\mathbf{0.800 \pm 0.006}$	$\mathbf{0.901 \pm 0.007}$	$\mathbf{0.952 \pm 0.005}$	$\mathbf{0.094 \pm 0.002}$	$\mathbf{0.005 \pm 0.002}$
S^2-PINN (no proj)						
Diffusion	$\mathbf{0.494 \pm 0.013}$	$\mathbf{0.796 \pm 0.015}$	$\mathbf{0.888 \pm 0.015}$	0.937 ± 0.012	$\mathbf{0.189 \pm 0.000}$	$\mathbf{0.014 \pm 0.008}$
Allen–Cahn	$\mathbf{0.496 \pm 0.013}$	$\mathbf{0.798 \pm 0.015}$	$\mathbf{0.890 \pm 0.015}$	$\mathbf{0.939 \pm 0.012}$	$\mathbf{0.189 \pm 0.000}$	$\mathbf{0.013 \pm 0.008}$
Burgers	$\mathbf{0.496 \pm 0.012}$	$\mathbf{0.798 \pm 0.014}$	$\mathbf{0.890 \pm 0.015}$	$\mathbf{0.938 \pm 0.011}$	$\mathbf{0.189 \pm 0.000}$	$\mathbf{0.013 \pm 0.008}$
Darcy	$\mathbf{0.501 \pm 0.007}$	$\mathbf{0.796 \pm 0.008}$	$\mathbf{0.894 \pm 0.007}$	$\mathbf{0.945 \pm 0.005}$	0.116 ± 0.000	$\mathbf{0.007 \pm 0.003}$
Vanilla PINN						
Diffusion	$\mathbf{0.494 \pm 0.027}$	$\mathbf{0.799 \pm 0.027}$	$\mathbf{0.900 \pm 0.019}$	$\mathbf{0.949 \pm 0.010}$	0.200 ± 0.007	0.017 ± 0.005
Allen–Cahn	$\mathbf{0.481 \pm 0.017}$	$\mathbf{0.783 \pm 0.029}$	$\mathbf{0.888 \pm 0.025}$	$\mathbf{0.940 \pm 0.023}$	0.199 ± 0.005	$\mathbf{0.018 \pm 0.020}$
Burgers	$\mathbf{0.499 \pm 0.014}$	$\mathbf{0.806 \pm 0.012}$	$\mathbf{0.903 \pm 0.005}$	$\mathbf{0.951 \pm 0.003}$	0.200 ± 0.005	$\mathbf{0.006 \pm 0.005}$
Darcy	0.450 ± 0.004	0.737 ± 0.013	0.848 ± 0.011	0.911 ± 0.007	0.093 ± 0.002	0.051 ± 0.008
MC-Dropout PINN						
Diffusion	0.420 ± 0.031	0.664 ± 0.024	0.750 ± 0.022	0.801 ± 0.017	0.211 ± 0.006	0.129 ± 0.023
Allen–Cahn	0.402 ± 0.017	0.671 ± 0.017	0.767 ± 0.017	0.822 ± 0.015	0.201 ± 0.006	0.122 ± 0.016
Burgers	0.390 ± 0.014	0.642 ± 0.027	0.736 ± 0.028	0.786 ± 0.026	0.212 ± 0.006	0.149 ± 0.024
Darcy	0.310 ± 0.020	0.569 ± 0.028	0.708 ± 0.036	0.811 ± 0.028	0.098 ± 0.003	0.188 ± 0.027
Deep Ensemble (K=5)						
Diffusion	$\mathbf{0.507 \pm 0.012}$	$\mathbf{0.818 \pm 0.022}$	$\mathbf{0.910 \pm 0.016}$	$\mathbf{0.954 \pm 0.010}$	$\mathbf{0.195 \pm 0.009}$	0.015 ± 0.008
Allen–Cahn	$\mathbf{0.495 \pm 0.011}$	$\mathbf{0.806 \pm 0.021}$	$\mathbf{0.904 \pm 0.017}$	$\mathbf{0.950 \pm 0.011}$	$\mathbf{0.195 \pm 0.009}$	0.013 ± 0.004
Burgers	0.530 ± 0.012	0.835 ± 0.021	0.920 ± 0.014	$\mathbf{0.959 \pm 0.009}$	0.196 ± 0.009	0.024 ± 0.013
Darcy	$\mathbf{0.498 \pm 0.008}$	$\mathbf{0.795 \pm 0.010}$	$\mathbf{0.894 \pm 0.009}$	$\mathbf{0.945 \pm 0.006}$	0.097 ± 0.001	$\mathbf{0.008 \pm 0.003}$

Computational cost. On stochastic diffusion, S^2-PINN trains in $\approx$90 mins with 1 GB peak memory, versus $\approx$80 mins /4 GB for the vanilla PINN; SPINN+Z and functional CP-PINN use less memory but are remarkably less accurate (Table 18; cost analysis in Appendix I).

Scaling stress test. On the diffusion scaling benchmark, S^2-PINN remains stable up to R=15 random dimensions, corresponding to G=816 gPC modes (Appendix G). Mean errors stay in the (4–7) $\times 10^{-3}$ and Cov@90% remains near 0.88. We further push the stochastic dimension to R=50 ($G=\binom{53}{3}$=23,426 modes), where S^2-PINN still attains $(1.09\pm0.14)\times10^{-2}$ mean and $(1.24\pm0.30)\times 10^{-2}$ variance relative L^2 error (Table 13). This provides evidence for scalability on this diffusion stress test, not as a universal high-dimensional guarantee. It is consistent with Theorem B.5, where the projection-estimation batch size depends logarithmically on G under boundedness assumptions.

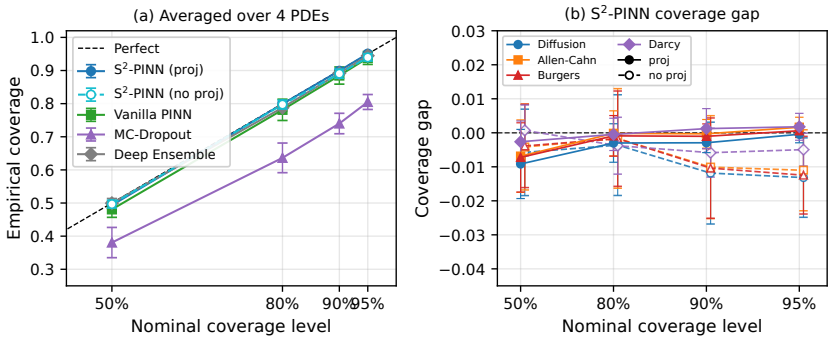

Figure 2: Reliability diagrams. (a) Empirical coverage vs. nominal level, averaged over 4 PDEs. S^2-PINN with projection and its no-projection ablation, Vanilla PINN; Deep Ensemble. The dashed diagonal indicates perfect calibration. (b) Per-PDE coverage gap, empirical minus nominal, for S^2-PINN with and without projection.

Calibration. Figure 2 evaluates predictive calibration through empirical coverage at nominal levels 50%, 80%, 90%, and 95%. Averaged across the four manufactured PDEs, S^2-PINN closely follows the ideal calibration diagonal, while MC-dropout consistently under-covers. The no-projection ablation remains reasonably calibrated at lower nominal levels but exhibits mild under-coverage at 90% and 95%, as made explicit by the coverage-gap plot in Fig. 2(b). This suggests that the projected residual is not merely improving mean and variance accuracy, but also helps stabilize the pushforward uncertainty intervals. Vanilla PINN and the deep ensemble can achieve near-nominal coverage on some controlled tasks, but their weaker moment accuracy indicates that calibration

alone is insufficient: a useful stochastic surrogate must match both the predictive intervals and the underlying solution statistics.

External validity: non-manufactured benchmarks. Manufactured solutions can provide artificially favorable optimization landscapes. To stress-test S^2-PINN beyond controlled settings, we evaluate on two non-manufactured forward problems with no closed-form solutions, using the same default hyperparameters as all manufactured benchmarks—no per-problem tuning.

(i) Stochastic Poisson source: $-\Delta u = f(x, y, \mathbf{Z})$ where the source f is a KL-expanded random field (σ=0.5), with u=0 on $\partial\Omega$. **(ii) Stochastic Darcy flow:** $-\nabla\cdot(k(x, y, \mathbf{Z})\nabla u) = 1$ where the permeability k is a log-normal field (σ=1.0), with u=0 on $\partial\Omega$. Both are solved on Ω=$[-1, 1]^2$ and evaluated against finite-difference Monte Carlo references (10,000 samples, 64×64 grid).

Table 4: Non-manufactured benchmark results. S^2-PINN uses the same default hyperparameters as all manufactured experiments; no per-problem tuning. Reference: FDM Monte Carlo (N=10,000, 64×64).

		Poisson MC (source)		Darcy MC (coefficient)	
Method	#Params	$\mathcal{E}_{\bar{u}}$	$\mathcal{E}_{\sigma^2}$	$\mathcal{E}_{\bar{u}}$	$\mathcal{E}_{\sigma^2}$
S^2-PINN	6 114	$\mathbf{0.335\pm0.122}$	$\mathbf{1.39\pm1.10}$	$\mathbf{0.74\pm0.23}$	15.4 ± 1.88
Vanilla PINN	199 937	19.7 ± 23.4	6.79 ± 6.82	1.40 ± 0.49	5.40 ± 1.92
Deep Ens. (K=5)	5×199 937	7.07 ± 1.11	77.8 ± 0.89	1.04 ± 0.19	$\mathbf{4.92 \pm 1.46}$

All errors $\times 10^{-2}$ (relative L^2).

On the Poisson problem, S^2-PINN achieves 59× lower mean error and 5× lower variance error than the vanilla PINN while using 33× fewer parameters. On the log-normal Darcy problem, S^2-PINN gives the best mean prediction (0.74%), but the deep ensemble gives the best variance (4.92%). This variance gap is a useful failure signal: low-rank gPC-aligned representations can under-resolve the nonlinear variance induced by exponential random coefficients.

Diagnosing the failure regime before training. The log-normal Darcy field violates the uniform boundedness (sub-Gaussian) assumption underlying Theorem B.5: the exponential map pushes substantial stochastic energy into high-degree modes, which leaks out of a low-degree subspace V_p. This can be diagnosed offline, before training, with the *subspace energy ratio*

$$\mathrm{SER}(p) = \frac{\sum_{|\boldsymbol{\alpha}|\le p} \|\hat{a}_{\boldsymbol{\alpha}}\|_{L^2}^2}{\mathbb{E}_{\mathbf{Z}}\left[\|a(\cdot, \mathbf{Z})\|_{L^2}^2\right]}, \tag{8}$$

i.e., the fraction of the energy of the input coefficient field $a(\mathbf{x}, \mathbf{Z})$ captured by its gPC coefficients $\hat{a}_{\boldsymbol{\alpha}}$ of total degree at most p. A value $\mathrm{SER}(p) < 95\%$ indicates that significant energy lies outside V_p, in which case the polynomial degree should be increased, a log-transformation applied, or the projection term disabled. On the log-normal Darcy problem, increasing the degree from p=2 to p=4 raises $\mathrm{SER}(p)$ above 95% and reduces the S^2-PINN variance error from 15.4% to 4.15%, below that of the deep ensemble (4.92%).

Stochastic Navier–Stokes. To test a harder, nonlinear vector-valued problem, we further consider a 2D stochastic incompressible Navier–Stokes problem in vorticity form with five uniform random variables controlling phase shifts, viscosity, and two vortex amplitudes (Appendix J). S^2-PINN attains velocity mean errors of about 2×10^{-3}, 5–17× lower than the baselines, with the lowest variance errors and the best calibration (Tables 15 and 16).

gPC projection is a regime-dependent regularizer. On diffusion, Allen–Cahn, and Burgers—all with polynomial dependence on $\mathbf{Z}$—the projection loss improves S^2-PINN's mean accuracy by 2–24%. On Darcy (exponential $\mathbf{Z}$-dependence), no-projection slightly outperforms projection on mean accuracy (9.51×10^{-4} vs. 1.05×10^{-3}). This is expected: the gPC projection penalizes residual components outside the polynomial subspace V_p, which is beneficial when the true solution is well-approximated by V_p (polynomial $\mathbf{Z}$) but can introduce bias when it is not (exponential $\mathbf{Z}$). Figure 3 further visualizes this regime-dependent behavior across different R values.

Projection in the vanilla PINN gives mixed results. Adding the gPC projection to a standard MLP-based PINN yields inconsistent behavior: it helps on Allen–Cahn (about 18% mean-error

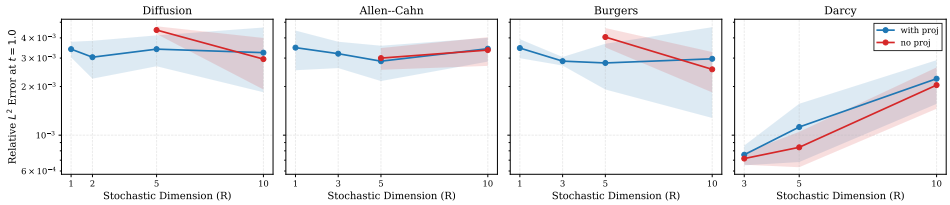

Figure 3: Regime map: relative L^2 at t=1.0 with/without gPC projection, across R values and PDEs (diffusion, Allen–Cahn, Burgers, Darcy). Shaded bands show mean ± std over 5 seeds. Projection consistently helps for polynomial stochastic dependence (diffusion/Allen–Cahn/Burgers, $R \leq 5$) and becomes neutral or detrimental at higher R or for non-polynomial/exponential $\mathbf{Z}$ (Darcy).

reduction) but degrades diffusion, Burgers, and Darcy. Critically, the improvement is smaller than in S^2-PINN, suggesting that the *separable tensor structure amplifies the benefit of Galerkin projection* by providing a representation that is structurally aligned with the gPC basis.

PC^2 comparison. We evaluate PC^2 on the manufactured diffusion benchmark and the two non-manufactured forward benchmarks using UQpy Olivier et al. [2020], fix p_Z=3, and choose the deterministic polynomial degree by held-out PDE/boundary residuals rather than test moment errors. Table 5 shows that physics-informed PCE is a strong but regime-sensitive baseline. On the smooth

Table 5: Physics-informed PCE (PC^2) comparison against S^2-PINN. Errors are raw relative L^2 values.

Problem	Solver	Selected p_{det}	# Basis	$\mathcal{E}_{\bar{u}}$	$\mathcal{E}_{\sigma^2}$
Poisson MC	LAR	6, 4, 4	15.67 ± 8.02	0.068 ± 0.078	0.957 ± 0.068
Poisson MC	OLS	8, 8, 8	2520 ± 0	$\mathbf{(1.59 \pm 0.05) \times 10^{-3}}$	$(1.44 \pm 0.39) \times 10^{-2}$
Poisson MC	S^2-PINN (proj)	–	–	$(3.35 \pm 1.22) \times 10^{-3}$	$\mathbf{(1.39 \pm 1.10) \times 10^{-2}}$
Diffusion	LAR	4, 4, 4	3.67 ± 1.15	1.103 ± 0.181	9.452 ± 14.640
Diffusion	OLS	3, 3, 3	1120 ± 0	25.798 ± 0.569	262520.941 ± 59229.735
Diffusion	S^2-PINN (proj)	–	–	$\mathbf{(3.71 \pm 1.01) \times 10^{-3}}$	$\mathbf{(4.40 \pm 1.52) \times 10^{-3}}$
Darcy MC	LAR	4, 4, 6	4.33 ± 2.08	0.844 ± 0.270	0.996 ± 0.006
Darcy MC	OLS	8, 8, 6	2202.67 ± 549.64	1.483 ± 0.349	1064.196 ± 81.247
Darcy MC	S^2-PINN (proj)	–	–	$\mathbf{(0.74 \pm 0.23) \times 10^{-2}}$	$\mathbf{(15.4 \pm 1.88) \times 10^{-2}}$

stochastic Poisson source, anisotropic PC^2 with OLS attains very low mean error and competitive variance error, confirming that global polynomial surrogates are effective when the solution is well captured by a low-order polynomial space. This strength does not transfer to the harder regimes: PC^2 becomes unstable on the transient diffusion benchmark and fails to recover log-normal Darcy variance, with sparse LAR selecting a nearly constant basis and dense OLS producing large variance errors. In contrast, S^2-PINN keeps the stochastic polynomial basis but replaces the global deterministic polynomial surrogate with learnable space–time features and low-rank coupling, yielding more stable physics-informed UQ across these regimes.

4.4 Stochastic inverse problems

Beyond forward UQ, the differentiable gPC structure of S^2-PINN naturally extends to *stochastic inverse problems*, where unknown PDE parameters must be inferred jointly with the solution field. We demonstrate this on two distinct benchmarks: the stochastic diffusion equation (linear) and the stochastic Burgers equation (nonlinear). For both PDEs, we consider a log-normal random field $k(\mathbf{x}, \mathbf{Z}) = \exp\big(\mu_k + \sigma \sum_{r=1}^{R} \sqrt{\lambda_r}\, \varphi_r(\mathbf{x})\, Z_r\big)$ (representing the diffusion coefficient and the kinematic viscosity, respectively), treating the log-mean μ_k and field standard deviation σ as unknown learnable parameters (true values: μ_k^*=0, σ^*=1). Starting from deliberately poor initial guesses ($\hat{\mu}_k$=0.5, $\hat{\sigma}$=0.5), we augment the physics loss with a data-fidelity term $\mathcal{L}_{\text{data}} = \frac{1}{N_{\text{obs}}} \sum_{i=1}^{N_{\text{obs}}} \|\hat{u}(t_i, \mathbf{x}_i, \mathbf{Z}_i) - u_i^{\text{obs}}\|^2$ computed over N_{obs}=1,000 scattered observations, weighted by w_{data}=100. The PDE parameters receive a 10× higher learning rate than the model weights via separate Adam parameter groups, enabling the two scalars to converge at a rate commensurate with the ∼7 k network weights. Table 6 summaries the results. For diffusion, both parameters are recovered to within 5×10^{-5} of their true values, and the solution field achieves mean relative L_2

error of $(1.29 \pm 0.36) \times 10^{-4}$ and variance relative L_2 error of $(1.06 \pm 0.48) \times 10^{-4}$. For Burgers, the nonlinear advection term creates a more challenging optimization landscape: the log-mean is recovered to within 10^{-3} and σ to within 5×10^{-4}, with mean solution error $(2.03 \pm 0.62) \times 10^{-3}$ and variance error $(3.71 \pm 0.87) \times 10^{-3}$—roughly one order of magnitude above diffusion, consistent with the harder forward-problem baseline. These results confirm that the gPC basis and CP factorization provide sufficient inductive bias for simultaneous parameter estimation and stochastic solution reconstruction across both linear and nonlinear regimes.

Table 6: Stochastic inverse problem results on the 2D diffusion and Burgers benchmarks. True parameters: μ_k^*=0.0, σ^*=1.0. Initial guesses: $\hat{\mu}_k$=0.5, $\hat{\sigma}$=0.5.

PDE	$\lvert\hat{\mu}_k - \mu_k^*\rvert$	$\lvert\hat{\sigma} - \sigma^*\rvert$	$\mathcal{E}_{\bar{u}}$ (Mean Rel. L_2)	$\mathcal{E}_{\sigma^2}$ (Var Rel. L_2)
Diffusion	$< 5 \times 10^{-5}$	$< 5 \times 10^{-5}$	$(1.29 \pm 0.36) \times 10^{-4}$	$(1.06 \pm 0.48) \times 10^{-4}$
Burgers	$< 1 \times 10^{-3}$	$< 5 \times 10^{-4}$	$(2.03 \pm 0.62) \times 10^{-3}$	$(3.71 \pm 0.87) \times 10^{-3}$

5 Conclusion

In this work, we have presented **S^2-PINN** (Stochastic Separable Physics-Informed Neural Network), an efficient framework that extends separable coordinate architectures to forward and inverse uncertainty quantification for random PDEs. By coupling learnable Gaussian spatial atoms, Fourier temporal features, and an orthonormal gPC stochastic basis via a low-rank CP tensor decomposition, S^2-PINN enables closed-form statistical moment evaluation and circumvents the curse of dimensionality inherent in dense space–time–stochastic grids. Our analysis also shows that S^2-PINN's gains come from these coupling three aligned ingredients. For polynomial-in-$\mathbf{Z}$ problems, the projection encourages residual consistency in the stochastic modes used by the model, and Theorem B.3 identifies the corresponding population objective with a stochastic Galerkin constraint. The theory further supports the main mechanism by specializing standard tools (tensor-product density, orthogonal projection, and Monte Carlo concentration) to the proposed architecture and establishing L^2 expressivity of finite-rank separable sums, mini-batch concentration for projected coefficients, and conditioning control for the learned spatial dictionary. Together, the theory and ablations support low-rank stochastic separability as an effective modeling principle for physics-informed neural UQ.

Limitations and future work. Although S^2-PINN shows great promise and outperforms across forward prediction, calibration, scaling, and inverse problems, there remain some frontiers for making S^2-PINN a more general random PDE solver. First, S^2-PINN currently inherits the regular-domain bias of separable coordinate architectures, and extending the method to complex geometries will require geometry-aware dictionaries, coordinate maps, or mesh-based variants. Second, our theoretical results establish asymptotic expressivity and characterize the projection estimator under stated assumptions, but not provide a finite-q, finite-G end-to-end solution-error guarantee: a finite rank q introduces low-rank approximation error and a finite G introduces stochastic truncation error. A promising way forward to overcome these limitations is to combine low-rank tensor representations with adaptive rank selection, learnable gPC truncation strategies to capture the distribution of random variables, and geometry-adaptive coordinate mappings.

Broader impact. Fast physics-informed UQ can support safer engineering design, climate and energy modeling, and medical-device simulation by making uncertainty estimates more accessible when repeated high-fidelity solves are expensive. The main risk is misuse through overconfident deployment: inaccurate calibration or unresolved variance in difficult stochastic regimes could lead to misleading confidence estimates. For this reason, real-world use should pair S^2-PINN with problem-specific validation, calibration checks, and conservative uncertainty reporting.

Acknowledgement

The authors gratefully acknowledge support from the U.S. Department of Energy (DOE grant DE-SC0025801).

References

H. Antil, S. Dolgov, and A. Onwunta. Ttrisk: Tensor train decomposition algorithm for risk-averse optimization. *NumericalLinear Algebra with Applications*, 3(3):e2481, 2023. URL `https://doi.org/10.1002/nla.2481.12`.

H. Antil, S. Dolgov, and A. Onwunta. Smoothed moreau-yosida tensor-train approximation of state-constrained optimization problems under uncertainty. *NumericalLinear Algebra with Applications*, page 32:e70028, 2025a. URL `https://doi.org/10.1002/nla.70028`.

H. Antil, S. Dolgov, and A. Onwunta. Tensor train solution to uncertain optimization problems with shared sparsity penalty. *Optimization and Engineering*, 2025b. URL `https://doi.org/10.1007/s11081-025-10036-7`.

I. Babuška, F. Nobile, and R. Tempone. A stochastic collocation method for elliptic partial differential equations with random input data. *SIAM Journal on Numerical Analysis*, 45(3):1005–1034, 2007.

M. Bachmayr. Low-rank tensor methods for partial differential equations. *Acta Numerica*, 32:1–121, 2023.

M. Bachmayr, R. Schneider, and A. Uschmajew. Tensor networks and hierarchical tensors for the solution of high-dimensional partial differential equations. *Foundations of Computational Mathematics*, 16:1423–1472, 2016.

B. Bahmani, I. G. Kevrekidis, and M. D. Shields. Neural chaos: A spectral stochastic neural operator. *Journal of Computational Physics*, 539:114233, 2025.

P. Benner, A. Onwunta, and M. Stoll. Low-rank solution of unsteady diffusion equations with stochastic coefficients. *SIAM/ASA Journal on Uncertainty Quantification*, 3(1):622–649, 2015.

P. Benner, S. Dolgov, A. Onwunta, and M. Stoll. Low-rank solvers for unsteady stokes–brinkman optimal control problem with random data. *Computer Methods in Applied Mechanics and Engineering*, 304:26–54, 2016a. ISSN 0045-7825. doi: https://doi.org/10.1016/j.cma.2016.02.004. URL `https://www.sciencedirect.com/science/article/pii/S0045782516300263`.

P. Benner, A. Onwunta, and M. Stoll. Block-diagonal preconditioning for optimal control problems constrained by PDEs with uncertain inputs. *SIAM Journal on Matrix Analysis and Applications*, 37(2):491–518, 2016b.

D. Bigoni, A. P. Engsig-Karup, and Y. M. Marzouk. Spectral tensor-train decomposition. *SIAM Journal on Scientific Computing*, 38(4):A2405–A2439, 2016.

F. Chinesta, A. Ammar, and E. Cueto. Recent advances and new challenges in the use of the proper generalized decomposition for solving multidimensional models. *Archives of Computational Methods in Engineering*, 17(4):327–350, 2010.

J. Cho, S. Nam, H. Yang, S.-B. Yun, Y. Hong, and E. Park. Separable physics-informed neural networks. In *Advances in Neural Information Processing Systems*, volume 36, pages 23761–23788, 2023.

A. Doostan, A. Validi, and G. Iaccarino. Non-intrusive low-rank separated approximation of high-dimensional stochastic models. *Computer Methods in Applied Mechanics and Engineering*, 263: 42–55, 2013.

V. A. Es'kin, D. V. Davydov, J. V. Gur'eva, A. O. Malkhanov, and M. E. Smorkalov. Separable physics-informed neural networks for the solution of elasticity problems. *arXiv preprint arXiv:2401.13486*, 2024.

Z. Fan, Y. Sun, S. Yang, and Y. Lu. Physics-informed inference time scaling for solving high-dimensional partial differential equations. In *International Conference on Learning Representations*, volume 2026, pages 127794–127844, 2026.

R. G. Ghanem and P. D. Spanos. *Stochastic Finite Elements: A Spectral Approach*. Springer, 1991.

S. Goswami, A. Bora, Y. Yu, and G. E. Karniadakis. Physics-informed deep neural operator networks. In *Machine learning in modeling and simulation: methods and applications*, pages 219–254. Springer, 2023.

J. Guo, X. Xie, C. Park, H. Zhang, M. J. Politis, G. Domel, and W. K. Liu. Interpolating neural network-tensor decomposition (inn-td): a scalable and interpretable approach for large-scale physics-based problems. In *International Conference on Machine Learning*, pages 21138–21162. PMLR, 2025.

G. E. Karniadakis, I. G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, and L. Yang. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, 2021.

T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM Review*, 51(3):455–500, 2009.

B. Lakshminarayanan, A. Pritzel, and C. Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 6402–6413, 2017. URL `https://proceedings.neurips.cc/paper/2017/hash/9ef2ed4b7fd2c810847ffa5fa85bce38-Abstract.html`.

Z. Li, N. B. Kovachki, K. Azizzadenesheli, K. Bhattacharya, A. Stuart, A. Anandkumar, et al. Fourier neural operator for parametric partial differential equations. In *International Conference on Learning Representations*, 2020.

Z. Liu, X. Yu, and Z. Zhang. TT-PINN: A tensor-compressed neural PDE solver for edge computing. *arXiv preprint arXiv:2207.01751*, 2022.

L. Lu, P. Jin, G. Pang, Z. Zhang, and G. E. Karniadakis. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3(3): 218–229, 2021.

A. Nouy. Proper generalized decompositions and separated representations for the numerical solution of high dimensional stochastic problems. *Archives of Computational Methods in Engineering*, 17 (4):403–434, 2010.

L. Novák, H. Sharma, and M. D. Shields. Physics-informed polynomial chaos expansions. *Journal of Computational Physics*, 506:112926, 2024.

A. Olivier, D. G. Giovanis, B. Aakash, M. Chauhan, L. Vandanapu, and M. D. Shields. Uqpy: A general purpose python package and development environment for uncertainty quantification. *Journal of Computational Science*, 47:101204, 2020.

I. V. Oseledets. Tensor-train decomposition. *SIAM Journal on Scientific Computing*, 33(5):2295–2317, 2011.

A. F. Psaros, X. Meng, Z. Zou, L. Guo, and G. E. Karniadakis. Uncertainty quantification in scientific machine learning: Methods, metrics, and comparisons. *Journal of Computational Physics*, 477: 111902, 2023.

M. Raissi, P. Perdikaris, and G. E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.

D. Shi, L. Lin, A. Han, L. Thompson, J. M. Hernández-Lobato, Z. Wang, and J. Gao. Expanding the chaos: Neural operator for stochastic (partial) differential equations. *arXiv preprint arXiv:2601.01021*, 2026.

C. Si and M. Yan. Initialization-enhanced physics-informed neural network with domain decomposition (IDPINN). *Journal of Computational Physics*, 530:113914, 2025.

C. Si, M. Yan, X. Li, and Z. Xia. Complex physics-informed neural network. *Journal of Computational Physics*, 553:114713, 2026.

S. K. Vemuri, T. Büchner, J. Niebling, and J. Denzler. Functional tensor decompositions for physics-informed neural networks. In *International Conference on Pattern Recognition*, pages 32–46. Springer, 2024.

D. Xiu and G. E. Karniadakis. The Wiener–Askey polynomial chaos for stochastic differential equations. *SIAM Journal on Scientific Computing*, 24(2):619–644, 2002.

L. Yang, X. Meng, and G. E. Karniadakis. B-PINNs: Bayesian physics-informed neural networks for forward and inverse PDE problems with noisy data. *Journal of Computational Physics*, 425: 109913, 2021.

X. Yang, L. Lu, and S. Mou. Derivative-informed training of neural operators on-the-fly via sketched tangent consistency, 2026. URL `https://arxiv.org/abs/2609.32797`.

P. Zanotta, L. Budinski, C. Aytekin, and V. Lahtinen. Quantum orthogonal separable physics-informed neural networks. *arXiv preprint arXiv:2511.12613*, 2025.

D. Zhang, L. Lu, L. Guo, and G. E. Karniadakis. Quantifying total uncertainty in physics-informed neural networks for solving forward and inverse stochastic problems. *Journal of Computational Physics*, 397:108850, 2019.

A Mathematical Formulation of Stochastic Elements and Tensor Factorization

This section details the mathematical construction of the stochastic fields, the chosen orthogonal polynomial basis, and the algebraic structure of the tensor decomposition that underpins S^2-PINN's efficiency.

A.1 Karhunen–Loève Expansion (KLE)

The stochastic coefficients (e.g., the diffusion field $k(x, y, \mathbf{Z})$) are parameterized using the Karhunen–Loève Expansion. The KLE provides an optimal mean-square representation of a stochastic process by decomposing it into an infinite linear combination of orthogonal spatial eigenfunctions multiplied by uncorrelated random variables.

For a random field defined by a specified covariance function $C(\mathbf{x}, \mathbf{x}')$, the KLE is given by

$$\log k(\mathbf{x}, \mathbf{Z}) = \mu_k + \sigma \sum_{i=1}^{\infty} \sqrt{\lambda_i}\phi_i(\mathbf{x})Z_i,$$

where μ_k is the mean of the log-field, σ controls the standard deviation (contrast), λ_i and $\phi_i(\mathbf{x})$ are the eigenvalues and eigenfunctions of the covariance operator $\int C(\mathbf{x}, \mathbf{x}')\phi_i(\mathbf{x}')d\mathbf{x}' = \lambda_i\phi_i(\mathbf{x})$, and $Z_i \sim \mathcal{N}(0, 1)$ are independent standard normal random variables.

In practical numerical implementations, this infinite series is truncated to the leading R terms that capture the vast majority of the field's variance. By treating $\mathbf{Z} = (Z_1, \ldots, Z_R)^\top \in \mathbb{R}^R$ as explicit inputs, the stochastic PDE is transformed into a deterministic, high-dimensional parametric PDE over the domain $\Omega \times \mathcal{T} \times \mathbb{R}^R$.

A.2 Generalized Polynomial Chaos (gPC)

The generalized Polynomial Chaos (gPC) framework provides a systematic approach to representing random processes through orthogonal polynomial expansions. For each probability distribution of the random input, there exists an optimal family of orthogonal polynomials that achieves *spectral convergence* in the L^2 sense. This correspondence, known as the *Wiener–Askey scheme*, is summarized in Table 7.

Table 7: Wiener–Askey scheme: optimal polynomial basis for common distributions.

Distribution	Support	Polynomial Family	Weight $w(z)$
Gaussian $\mathcal{N}(0, 1)$	$(-\infty, \infty)$	Hermite	$e^{-z^2/2}/\sqrt{2\pi}$
Uniform $\mathcal{U}(-1, 1)$	$[-1, 1]$	Legendre	$1/2$
Exponential $\mathrm{Exp}(1)$	$[0, \infty)$	Laguerre	e^{-z}
Beta $\mathrm{Beta}(\alpha, \beta)$	$[0, 1]$	Jacobi $P_n^{(\alpha,\beta)}$	$z^{\beta-1}(1-z)^{\alpha-1}$
Gamma $\Gamma(\alpha, 1)$	$[0, \infty)$	Gen. Laguerre $L_n^{(\alpha)}$	$z^{\alpha-1}e^{-z}$

In our implementation, S^2-PINN automatically selects the polynomial family from this table based on the marginal distribution of each random input Z_i. For problems with mixed distributions (e.g., some Z_i Gaussian and others uniform), the multivariate basis is formed as a tensor product of the corresponding 1D families.

Multivariate basis and total-degree truncation. For the R-dimensional vector $\mathbf{Z}$, the multidimensional gPC basis is constructed via the tensor product of 1D polynomials, indexed by a multi-index $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_R) \in \mathbb{N}_0^R$

$$\Psi_{\boldsymbol{\alpha}}(\mathbf{Z}) = \prod_{i=1}^{R} \psi_{\alpha_i}(Z_i),$$

where ψ_{α_i} denotes the degree-α_i polynomial from the family corresponding to the marginal distribution of Z_i (e.g., P_{α_i} for uniform). To avoid an uncontrolled combinatorial explosion, we employ

a *total degree truncation* scheme, limiting the basis to all polynomials where the sum of degrees $|\boldsymbol{\alpha}| = \sum_{i=1}^{R} \alpha_i \leq p$. This truncation drastically sparsifies the tensor product space. By a standard combinatorial "stars and bars" argument, finding the number of non-negative integer tuples $(\alpha_1, \ldots, \alpha_R)$ satisfying $\sum_{i=1}^{R} \alpha_i \leq p$ is equivalent to distributing up to p indistinguishable degrees among R distinct variables. By introducing a "slack" variable to absorb the remaining $p - \sum_{i=1}^{R} \alpha_i$ degrees, this equals the number of ways to distribute exactly p items among $R+1$ bins, yielding exactly $G = \binom{R+p}{p}$ active gPC modes. To illustrate how this basis is explicitly constructed, consider a 2D random space (R=2) with maximum degree p=3 and uniform inputs. The total number of valid modes is $G = \binom{2+3}{3} = 10$. The valid multi-indices (α_1, α_2) and their corresponding 2D Legendre polynomials $\Psi_{(\alpha_1,\alpha_2)}(Z_1, Z_2) = H_{\alpha_1}(Z_1)H_{\alpha_2}(Z_2)$ are systematically enumerated by their total degree $|\boldsymbol{\alpha}|$:

- **Degree 0:** $(0,0) \Rightarrow 1$
- **Degree 1:** $(1,0) \Rightarrow Z_1,\ (0,1) \Rightarrow Z_2$
- **Degree 2:** $(2,0) \Rightarrow \frac{1}{2}(3Z_1^2 - 1),\ (1,1) \Rightarrow Z_1 Z_2,\ (0,2) \Rightarrow \frac{1}{2}(3Z_2^2 - 1)$
- **Degree 3:** $(3,0) \Rightarrow \frac{1}{2}(5Z_1^3 - 3Z_1),\ (2,1) \Rightarrow \frac{1}{2}(3Z_1^2 - 1)Z_2,\ (1,2) \Rightarrow Z_1 \frac{1}{2}(3Z_2^2 - 1),\ (0,3) \Rightarrow \frac{1}{2}(5Z_2^3 - 3Z_2)$

This targeted truncation avoids the exponential blow-up of a full tensor grid. For higher dimensions such as R=10 and p=3, we only need G=286 modes instead of the naïve $4^{10} \approx 10^6$ grid points required by full tensor products.

A.3 Canonical Polyadic (CP) Tensor Decomposition vs. Tensor Train (TT)

The core architecture of S^2-PINN decomposes the multidimensional solution $u(\mathbf{x}, t, \mathbf{Z})$ into a rank-q Canonical Polyadic (CP) tensor product.

If we represent the network's internal features as an abstract 3-way tensor $\mathcal{W} \in \mathbb{R}^{M \times M_t \times G}$, computing the full dense tensor would require $\mathcal{O}(MM_tG)$ parameters. The CP decomposition factors this tensor into a sum of q rank-1 components

$$u(\mathbf{x}, t, \mathbf{Z}) = \sum_{r=1}^{q} w_r \left(\mathbf{A}_r^\top \varphi(\mathbf{x})\right) \left(\mathbf{B}_r^\top \psi(t)\right) \left(\mathbf{D}_r^\top \Phi(\mathbf{Z})\right),$$

where $\mathbf{A} \in \mathbb{R}^{M \times q}$, $\mathbf{B} \in \mathbb{R}^{M_t \times q}$, and $\mathbf{D} \in \mathbb{R}^{G \times q}$ are factor matrices, and $\mathbf{w} \in \mathbb{R}^q$ is a learnable scaling vector. This reduces the parameter complexity from multiplicative to additive: $\mathcal{O}(q(M + M_t + G))$.

Why CP instead of Tensor Train (TT)? While Tensor Train is a powerful decomposition for ultra-high dimensional spaces ($D \gg 10$), our functional representation operates fundamentally on a 3-way tensor (Space $\times$ Time $\times$ Stochastic), because the spatial coordinates (x, y) are encoded together by the spatial dictionary and the R-dimensional $\mathbf{Z}$ is encoded together by the gPC basis. For a 3-way tensor, CP provides several decisive advantages over TT for PINN optimization:

1. **Decoupled Autodiff:** In CP, the spatial derivative operators (e.g., the Laplacian Δ) distribute purely across the sum and act *only* on the spatial terms $\mathbf{A}_r^\top \varphi(\mathbf{x})$. The time and stochastic components act as constants. This means the computational cost of automatic differentiation does not scale with G. In a TT format, computing spatial derivatives requires backpropagating through a chain of matrix multiplications, which introduces significant memory overhead and stiffness.
2. **Explicit Galerkin Coefficients:** Our hybrid loss requires computing the exact Galerkin projection of the PDE residual onto the gPC modes $\Phi_g(\mathbf{Z})$. In the CP formulation, the macroscopic scalar coefficient attached to any gPC mode Φ_g is explicitly given by the scalar sum $\sum_{r=1}^{q} w_r (\mathbf{A}_r^\top \varphi(\mathbf{x}))(\mathbf{B}_r^\top \psi(t)) D_{g,r}$. In a TT format, the stochastic subspace is folded inside a core matrix, making it algebraically opaque and computationally expensive to extract exact projection coefficients.

B Theoretical Analysis

In this section, we collect PDE-agnostic results that support the main design choices in S^2-PINN: finite-rank separable expressivity, projection–Galerkin consistency, mini-batch stability of projected residual coefficients, orthogonality conditioning regularization, and the algebraic efficiency of the CP coupling. Instead of proving optimization convergence or finite-rank rates for a specific PDE, these results formalize the mechanism used by the architecture and loss. We emphasize that this section does not introduce new abstract functional-analytic tools: the arguments rely on standard results (density of algebraic tensor products, orthogonal projection and Parseval identities, and Monte Carlo concentration). Their method-specific content is twofold. (i) *Expressive completeness*: Theorem B.2 shows that a low-rank CP coupling of spatio-temporal features with a gPC basis preserves universal approximation in L^2, so the factorization introduces no representation bottleneck in the limit. (ii) *Error decomposition*: Theorems B.2–B.4 separate the total error into CP-rank approximation error (controlled by q), gPC truncation error (controlled by p), and Monte Carlo sampling error (controlled by the batch size B), which guides the balance between network capacity, spectral degree, and sampling budget.

Remark B.1 (Scope of the guarantees). The results below are asymptotic or estimator-level statements. They do not provide a finite-q, finite-G end-to-end bound on the solution error of a trained model: a finite rank q incurs low-rank approximation error, a finite G incurs stochastic truncation error, and non-convex optimization under a finite budget is not covered.

B.1 Setup and notation

Let $(\mathcal{T}, \mu_t)$, (Ω, μ_x), and $(\mathbf{Z}, \mathbb{P}_{\mathbf{Z}})$ be measure spaces for time, spatial, and stochastic, respectively, and $L^2(\mathcal{T})$, $L^2(\Omega)$, $L^2(\mathbf{Z})$ are corresponding L^2 spaces. Define the product space

$$\mathcal{H} := L^2(\mathcal{T}) \otimes L^2(\Omega) \otimes L^2(\mathbf{Z})$$

with norm $\|\cdot\|_{L^2(\mathcal{T}\times\Omega\times\mathbf{Z})}$. Let $\{T_k\}_{k=1}^{M_t} \subset L^2(\mathcal{T})$, $\{\Phi_i\}_{i=1}^{M} \subset L^2(\Omega)$, $\{\Psi_g\}_{g=1}^{G} \subset L^2(\mathbf{Z})$ be (not necessarily orthonormal) feature families. Throughout, we make explicit the *mild density conditions* invoked in the main body: the feature families are total in their respective L^2 spaces, i.e. $\overline{\mathrm{span}}\{T_k\}_{k\geq 1} = L^2(\mathcal{T})$, $\overline{\mathrm{span}}\{\Phi_i\}_{i\geq 1} = L^2(\Omega)$, and $\overline{\mathrm{span}}\{\Psi_g\}_{g\geq 1} = L^2(\mathbf{Z})$, so that the closures $\mathcal{H}_{\mathrm{time}}$, $\mathcal{H}_{\mathrm{space}}$, and $\mathcal{H}_{\mathrm{stoc}}$ introduced in Theorem B.2 below coincide with the full L^2 spaces. This assumption is harmless for the bases used in our experiments (Hermite/Legendre polynomials in $\mathbf{Z}$ are total in $L^2(\mathbf{Z})$, and our spatial Gaussian dictionary becomes total in $L^2(\Omega)$ in the limit $M \to \infty$). For a generic signal $f : \mathcal{T} \times \Omega \times \mathbf{Z} \to \mathbb{R}$, define its (strong-form) "residual" symbolically as $r := \mathcal{R}[f]$ where $\mathcal{R}$ is any measurable operator. None of the results below require a PDE structure of $\mathcal{R}$.

B.2 Expressivity of separable sums

Let $\mathcal{H}_{\mathrm{space}} = \overline{\mathrm{span}}\{\Phi_i\}_{i\geq 1} \subset L^2(\Omega)$, $\mathcal{H}_{\mathrm{time}} = \overline{\mathrm{span}}\{T_k\}_{k\geq 1} \subset L^2(\mathcal{T})$, and $\mathcal{H}_{\mathrm{stoc}} = \overline{\mathrm{span}}\{\Psi_g\}_{g\geq 1} \subset L^2(\mathbf{Z})$ be dense subspaces (closures in L^2). Define the rank-q separable class

$$\mathcal{F}_q := \Big\{ \sum_{r=1}^{q} \tau^{(r)}(t)\, \phi^{(r)}(\mathbf{x})\, \psi^{(r)}(\mathbf{z}) :\ \tau^{(r)} \in \mathcal{H}_{\mathrm{time}},\, \phi^{(r)} \in \mathcal{H}_{\mathrm{space}},\ \psi^{(r)} \in \mathcal{H}_{\mathrm{stoc}} \Big\}.$$

Theorem B.2 (Density of finite-rank separable sums)**.** *The union $\bigcup_{q\geq 1} \mathcal{F}_q$ is dense in the Hilbert tensor product $\mathcal{H}_{\mathrm{time}} \otimes \mathcal{H}_{\mathrm{space}} \otimes \mathcal{H}_{\mathrm{stoc}} \subset L^2(\mathcal{T} \times \Omega \times \mathbf{Z})$.*

Proof. Define the algebraic tensor product of these three spaces, denoted as

$$\mathcal{V} = \mathcal{H}_{\mathrm{time}} \otimes_{\mathrm{alg}} \mathcal{H}_{\mathrm{space}} \otimes_{\mathrm{alg}} \mathcal{H}_{\mathrm{stoc}}.$$

The elements of the algebraic tensor product $\mathcal{V}$ are strictly defined as formal finite linear combinations. That is, any element $f \in \mathcal{V}$ can be expressed as

$$f(t, \mathbf{x}, \mathbf{z}) = \sum_{r=1}^{q} \tau^{(r)}(t)\, \phi^{(r)}(\mathbf{x})\, \psi^{(r)}(\mathbf{z}),$$

where $q \in \mathbb{N}$ is a finite positive integer, and $\tau^{(r)} \in \mathcal{H}_{\mathrm{time}}$, $\phi^{(r)} \in \mathcal{H}_{\mathrm{space}}$, $\psi^{(r)} \in \mathcal{H}_{\mathrm{stoc}}$.

By definition, the function class $\mathcal{F}_q$ of separable functions of rank q precisely describes the set of functions formed by the addition of at most q pure tensors. Therefore, taking the union over all possible finite ranks $q \geq 1$ exactly yields the entire algebraic tensor product space $\bigcup_{q\geq 1} \mathcal{F}_q = \mathcal{V}$.

For two tensors $u_1 \otimes v_1 \otimes w_1$ and $u_2 \otimes v_2 \otimes w_2$, their inner product is defined as the product of the inner products within their respective spaces

$$\langle u_1 \otimes v_1 \otimes w_1, u_2 \otimes v_2 \otimes w_2 \rangle = \langle u_1, u_2 \rangle_{\mathcal{H}_{\text{time}}} \langle v_1, v_2 \rangle_{\mathcal{H}_{\text{space}}} \langle w_1, w_2 \rangle_{\mathcal{H}_{\text{stoc}}}.$$

Since all three spaces are subspaces of L^2 spaces, the aforementioned inner product is equivalent to the standard L^2 inner product under the product measure $\mu_t \otimes \mu_x \otimes \mathbb{P}_{\mathbf{Z}}$.

Extending this inner product bilinearly to the entire algebraic tensor product $\mathcal{V}$ induces a norm $\|\cdot\|_{L^2(\mathcal{T}\times\Omega\times\mathbf{Z})}$. We denote the Hilbert tensor product as $\mathcal{H}_{\text{time}} \otimes \mathcal{H}_{\text{space}} \otimes \mathcal{H}_{\text{stoc}}$. By definition, it is the completion (or closure) of the algebraic tensor product $\mathcal{V}$ with respect to this induced L^2 norm

$$\mathcal{H}_{\text{time}} \otimes \mathcal{H}_{\text{space}} \otimes \mathcal{H}_{\text{stoc}} := \overline{\mathcal{V}}^{\|\cdot\|_{L^2}}.$$

According to the fundamental principles of topology and metric spaces, the closure of a set is the smallest closed set containing it. Furthermore, a subset $\mathcal{V}$ is dense in a space $\mathcal{H}$ if and only if the closure of $\mathcal{V}$ is exactly $\mathcal{H}$.

Given that the Hilbert tensor product is intrinsically the closure of the algebraic tensor product, we have

$$\overline{\bigcup_{q\geq 1} \mathcal{F}_q}^{\|\cdot\|_{L^2}} = \overline{\mathcal{V}}^{\|\cdot\|_{L^2}} = \mathcal{H}_{\text{time}} \otimes \mathcal{H}_{\text{space}} \otimes \mathcal{H}_{\text{stoc}}.$$

This establishes that for any arbitrary function f in the Hilbert tensor product space and any given error tolerance $\epsilon > 0$, there necessarily exists a sufficiently large finite integer q and an approximant $\tilde{f} = \sum_{r=1}^{q} \phi^{(r)} \tau^{(r)} \psi^{(r)} \in \mathcal{F}_q$ such that:

$$\|f - \tilde{f}\|_{L^2(\mathcal{T}\times\Omega\times\mathbf{Z})} < \epsilon.$$

Consequently, the union of finite-rank separable functions, $\bigcup_{q\geq 1} \mathcal{F}_q$, is dense in the Hilbert tensor product space $\mathcal{H}_{\text{time}} \otimes \mathcal{H}_{\text{space}} \otimes \mathcal{H}_{\text{stoc}}$.

Finally, under the mild density conditions stated in Section B, the closures $\mathcal{H}_{\text{time}}$, $\mathcal{H}_{\text{space}}$, $\mathcal{H}_{\text{stoc}}$ coincide with the full L^2 spaces, so the Hilbert tensor product satisfies

$$\mathcal{H}_{\text{time}} \otimes \mathcal{H}_{\text{space}} \otimes \mathcal{H}_{\text{stoc}} = L^2(\mathcal{T}) \otimes L^2(\Omega) \otimes L^2(\mathbf{Z}) \cong L^2(\mathcal{T} \times \Omega \times \mathbf{Z}),$$

where the final isomorphism is the standard one for σ-finite product measures (a consequence of Fubini's theorem). Combining this identification with the density just established yields

$$\overline{\bigcup_{q\geq 1} \mathcal{F}_q}^{\|\cdot\|_{L^2}} = L^2(\mathcal{T} \times \Omega \times \mathbf{Z}).$$

□

B.3 Projection in the random space

Assume $\{\Psi_g\}_{g=1}^{G}$ are orthonormal in $L^2(\mathbf{Z})$ and let $V_p = \operatorname{span}\{\Psi_g\}_{g=1}^{G}$.

Theorem B.3 (Projection–Galerkin equivalence in $L^2(\mathbf{Z})$)**.** *For any measurable $h : \mathbf{Z} \to \mathbb{R}$,*

$$\sum_{g=1}^{G} \left(\mathbb{E}[\, h(\mathbf{Z})\, \Psi_g(\mathbf{Z})\,]\right)^2 = \left\|\mathsf{P}_{V_p} h\right\|^2_{L^2(\mathbf{Z})},$$

$$\mathbb{E}[\, h\Psi_g\,] = 0 \;\forall g \iff \mathsf{P}_{V_p} h = 0.$$

Proof. According to the Orthogonal Projection Theorem in Hilbert spaces, for any function $h \in L^2(\mathbf{Z})$ can be uniquely decomposed into an orthogonal sum:

$$h = h_{\parallel} + h_{\perp},$$

where $h_{\parallel} = \mathsf{P}_{V_p} h \in V_p$ is the orthogonal projection of h onto the subspace V_p, and $h_{\perp} \in V_p^{\perp}$ is the residual component such that $\langle h_{\perp}, v \rangle_{L^2(\mathbf{Z})} = 0$ for all $v \in V_p$. Given that $V_p = \operatorname{span}\{\Psi_g\}_{g=1}^{G}$, the projection $h_{\parallel}$ can be expressed as a linear combination of the basis functions

$$h_{\parallel} = \sum_{g=1}^{G} \alpha_g \Psi_g,$$

where $\alpha_g \in \mathbb{R}$ denote the expansion coefficients.

Part I: Proof of the First Equality. In the $L^2(\mathbf{Z})$ space, the inner product is equivalent to the expectation under the probability measure $\mathbb{P}_{\mathbf{Z}}$, i.e., $\langle u, v \rangle_{L^2} = \mathbb{E}[uv]$. Taking the inner product of h with the g'-th basis function $\Psi_{g'}$, we have

$$\mathbb{E}[\, h\, \Psi_{g'}\,] = \langle h, \Psi_{g'} \rangle_{L^2(\mathbf{Z})}.$$

Substituting the orthogonal decomposition $h = \sum_{g=1}^{G} \alpha_g \Psi_g + h_{\perp}$ into the expression above yields

$$\mathbb{E}[\, h\, \Psi_{g'}\,] = \left\langle \sum_{g=1}^{G} \alpha_g \Psi_g + h_{\perp}, \Psi_{g'} \right\rangle_{L^2(\mathbf{Z})} = \sum_{g=1}^{G} \alpha_g \langle \Psi_g, \Psi_{g'} \rangle_{L^2(\mathbf{Z})} + \langle h_{\perp}, \Psi_{g'} \rangle_{L^2(\mathbf{Z})}.$$

We now invoke the following properties:

1. Since $h_{\perp} \perp V_p$ and $\Psi_{g'} \in V_p$, it follows that $\langle h_{\perp}, \Psi_{g'} \rangle_{L^2(\mathbf{Z})} = 0$.
2. The basis $\{\Psi_g\}$ is orthonormal in $L^2(\mathbf{Z})$, meaning $\langle \Psi_g, \Psi_{g'} \rangle_{L^2(\mathbf{Z})} = \delta_{g,g'}$, where $\delta_{g,g'}$ is the Kronecker delta.

Substituting these properties, all terms in the summation vanish except for $g = g'$, leading to $\mathbb{E}[\, h\, \Psi_{g'}\,] = \alpha_{g'}$. This confirms that the projection coefficients are precisely the expectations (inner products) of h with the basis functions.

Next, we compute the squared L^2-norm of the projection $h_{\parallel} = \mathsf{P}_{V_p} h$,

$$\left\| \mathsf{P}_{V_p} h \right\|_{L^2(\mathbf{Z})}^2 = \left\| \sum_{g=1}^{G} \alpha_g \Psi_g \right\|_{L^2(\mathbf{Z})}^2.$$

Applying the orthonormality of $\{\Psi_g\}$ (a generalized form of the Pythagorean theorem or Parseval's identity), we obtain

$$\left\| \mathsf{P}_{V_p} h \right\|_{L^2(\mathbf{Z})}^2 = \sum_{g=1}^{G} \alpha_g^2.$$

Substituting $\alpha_g = \mathbb{E}[\, h(\mathbf{Z})\, \Psi_g(\mathbf{Z})\,]$ into the equation above directly yields the first desired identity

$$\sum_{g=1}^{G} \left(\mathbb{E}[\, h(\mathbf{Z})\, \Psi_g(\mathbf{Z})\,] \right)^2 = \left\| \mathsf{P}_{V_p} h \right\|_{L^2(\mathbf{Z})}^2.$$

Part II: Proof of the Equivalence. We aim to prove that $\mathbb{E}[\, h \Psi_g\,] = 0$ for all $g \in \{1, \ldots, G\}$ if and only if $\mathsf{P}_{V_p} h = 0$. From Part I, we have established

$$\left\| \mathsf{P}_{V_p} h \right\|_{L^2(\mathbf{Z})}^2 = \sum_{g=1}^{G} \left(\mathbb{E}[\, h \Psi_g\,] \right)^2.$$

Sufficiency ($\Longrightarrow$): If $\mathbb{E}[\, h\Psi_g\,] = 0$ for all $g \in \{1, \ldots, G\}$, then every term in the summation on the right-hand side is zero. Consequently, $\left\|\mathsf{P}_{V_p} h\right\|^2_{L^2(\mathbf{Z})} = 0$. By the definiteness property of the norm, it follows that $\mathsf{P}_{V_p} h = 0$.

Necessity ($\Longleftarrow$): Conversely, if $\mathsf{P}_{V_p} h = 0$, then its squared norm must be zero, $\left\|\mathsf{P}_{V_p} h\right\|^2_{L^2(\mathbf{Z})} = 0$. This implies $\sum_{g=1}^{G} \left(\mathbb{E}[\, h\Psi_g\,]\right)^2 = 0$. Since this is a sum of non-negative real terms (squares), the sum equals zero if and only if each individual term is zero. Therefore, it must hold that $\mathbb{E}[\, h\Psi_g\,] = 0$ for all $g \in \{1, \ldots, G\}$. □

B.4 Mini-batch estimator and uniform concentration

The two theorems in this subsection control the Monte-Carlo error incurred when the projection coefficients $\mathbb{E}[h\Psi_g]$ are estimated from a finite collocation batch. The first establishes unbiasedness and the standard $1/B$ variance scaling and the second upgrades this to a high-probability uniform bound across all G basis functions, which is the form used in our training analysis.

Theorem B.4 (Mini-batch estimator: unbiasedness and variance)**.** *Let* $\widehat{m}_g = \frac{1}{B}\sum_{b=1}^{B} h(\mathbf{Z}_b)\Psi_g(\mathbf{Z}_b)$ *with i.i.d.* $\mathbf{Z}_b \sim \mathbb{P}_{\mathbf{Z}}$*. Then* $\mathbb{E}\widehat{m}_g = \mathbb{E}[h\Psi_g]$ *and* $\mathrm{Var}(\widehat{m}_g) = \mathrm{Var}(h\Psi_g)/B$*. Moreover, for* $\widehat{\mathbf{m}} = (\widehat{m}_g)_{g=1}^{G}$ *and* $\mathbf{m} = (\mathbb{E}[h\Psi_g])_{g=1}^{G}$*,* $\mathbb{E}\|\widehat{\mathbf{m}}\|_2^2 = \|\mathbf{m}\|_2^2 + \frac{1}{B}\sum_{g=1}^{G} \mathrm{Var}(h\Psi_g)$*.*

Proof. **Part I: Unbiasedness of the Estimator $\mathbb{E}[\widehat{m}_g] = \mathbb{E}[\, h\Psi_g\,]$.** By definition, the estimator is given by

$$\widehat{m}_g = \frac{1}{B}\sum_{b=1}^{B} h(\mathbf{Z}_b)\Psi_g(\mathbf{Z}_b).$$

Taking the expectation of $\widehat{m}_g$ and invoking the linearity of expectation, we have

$$\mathbb{E}[\widehat{m}_g] = \mathbb{E}\left[\frac{1}{B}\sum_{b=1}^{B} h(\mathbf{Z}_b)\Psi_g(\mathbf{Z}_b)\right] = \frac{1}{B}\sum_{b=1}^{B} \mathbb{E}[\, h(\mathbf{Z}_b)\Psi_g(\mathbf{Z}_b)\,].$$

Since the samples $\{\mathbf{Z}_b\}_{b=1}^{B}$ are independent and identically distributed (i.i.d.) following the probability measure $\mathbb{P}_{\mathbf{Z}}$, the expectation for each b equals the population expectation

$$\mathbb{E}[\, h(\mathbf{Z}_b)\Psi_g(\mathbf{Z}_b)\,] = \mathbb{E}[\, h(\mathbf{Z})\Psi_g(\mathbf{Z})\,] = m_g.$$

Substituting this back into the expression yields

$$\mathbb{E}[\widehat{m}_g] = \frac{1}{B}\sum_{b=1}^{B} m_g = \frac{1}{B}(B \cdot m_g) = m_g = \mathbb{E}[\, h\Psi_g\,].$$

This confirms that $\widehat{m}_g$ is an unbiased estimator of m_g.

Part II: Variance of the Estimator $\mathrm{Var}(\widehat{m}_g) = \mathrm{Var}(h\Psi_g)/B$. We now consider the variance of $\widehat{m}_g$. Recall that for independent random variables $\{X_i\}$, $\mathrm{Var}(\sum X_i) = \sum \mathrm{Var}(X_i)$, and for a constant C, $\mathrm{Var}(CX) = C^2\mathrm{Var}(X)$. Applying these properties, we have

$$\mathrm{Var}(\widehat{m}_g) = \mathrm{Var}\left(\frac{1}{B}\sum_{b=1}^{B} h(\mathbf{Z}_b)\Psi_g(\mathbf{Z}_b)\right) = \frac{1}{B^2}\sum_{b=1}^{B} \mathrm{Var}\big(h(\mathbf{Z}_b)\Psi_g(\mathbf{Z}_b)\big).$$

Because the samples $\mathbf{Z}_b$ are identically distributed, the variance is constant for all b,

$$\mathrm{Var}\big(h(\mathbf{Z}_b)\Psi_g(\mathbf{Z}_b)\big) = \mathrm{Var}(h\Psi_g).$$

Substituting this back, we obtain

$$\mathrm{Var}(\widehat{m}_g) = \frac{1}{B^2}\big(B \cdot \mathrm{Var}(h\Psi_g)\big) = \frac{\mathrm{Var}(h\Psi_g)}{B}.$$

This demonstrates that the variance of the estimator decays at the standard Monte Carlo rate of $O(1/B)$ relative to the batch size B.

Part III: Expected Squared L_2 Norm $\mathbb{E}\|\widehat{\mathbf{m}}\|_2^2 = \|\mathbf{m}\|_2^2 + \frac{1}{B}\sum_{g=1}^{G}\mathrm{Var}(h\Psi_g)$. Using the definition of the squared L_2 norm and applying the linearity of expectation, we obtain

$$\mathbb{E}\|\widehat{\mathbf{m}}\|_2^2 = \mathbb{E}\Big[\sum_{g=1}^{G}\widehat{m}_g^2\Big] = \sum_{g=1}^{G}\mathbb{E}[\widehat{m}_g^2].$$

Utilizing the fundamental relation $\mathbb{E}[X^2] = (\mathbb{E}[X])^2 + \mathrm{Var}(X)$, we can decompose the second moment of each coefficient and get

$$\mathbb{E}[\widehat{m}_g^2] = (\mathbb{E}[\widehat{m}_g])^2 + \mathrm{Var}(\widehat{m}_g).$$

Substituting the unbiasedness result ($\mathbb{E}[\widehat{m}_g] = m_g$) and the variance result ($\mathrm{Var}(\widehat{m}_g) = \frac{1}{B}\mathrm{Var}(h\Psi_g)$) derived previously, we have

$$\mathbb{E}[\widehat{m}_g^2] = m_g^2 + \frac{1}{B}\mathrm{Var}(h\Psi_g).$$

Summing over all $g \in \{1, \ldots, G\}$ gives

$$\mathbb{E}\|\widehat{\mathbf{m}}\|_2^2 = \sum_{g=1}^{G} m_g^2 + \frac{1}{B}\sum_{g=1}^{G}\mathrm{Var}(h\Psi_g).$$

Recognizing that $\|\mathbf{m}\|_2^2 = \sum_{g=1}^{G} m_g^2$ is the squared norm of the true coefficient vector, we obtain

$$\mathbb{E}\|\widehat{\mathbf{m}}\|_2^2 = \|\mathbf{m}\|_2^2 + \frac{1}{B}\sum_{g=1}^{G}\mathrm{Var}(h\Psi_g).$$

□

Theorem B.5 (Uniform concentration over g)**.** *If $\{h(\mathbf{Z})\Psi_g(\mathbf{Z})\}_{g\le G}$ are sub-Gaussian with common proxy σ^2, then for any $\epsilon, \delta \in (0,1)$, if $B \gtrsim \sigma^2\epsilon^{-2}\log(G/\delta)$,*

$$\mathbb{P}\Big(\max_{1\le g\le G}\big|\widehat{m}_g - \mathbb{E}[h\Psi_g]\big| \le \epsilon\Big) \ge 1-\delta.$$

Proof. Since the samples $\mathbf{Z}_b$ are independent and identically distributed (i.i.d.), the sequence of random variables $\{X_g(\mathbf{Z}_b)\}_{b=1}^{B}$ are mutually independent, each being a sub-Gaussian random variable with proxy variance σ^2.

By the properties of sub-Gaussian random variables, the empirical mean $\widehat{m}_g$ of independent sub-Gaussian variables is also sub-Gaussian. Specifically, the proxy variance of the mean $\widehat{m}_g$ scales by a factor of $1/B$ relative to the single-sample proxy variance, given by $\sigma^2_{\text{mean}} = \frac{\sigma^2}{B}$. Applying the Hoeffding-type tail bound for centered sub-Gaussian random variables, the probability that the empirical mean deviates from the true mean m_g by more than ϵ satisfies

$$\mathbb{P}\Big(\big|\widehat{m}_g - m_g\big| > \epsilon\Big) \le 2\exp\left(-\frac{\epsilon^2}{2\sigma^2_{\text{mean}}}\right) = 2\exp\left(-\frac{B\epsilon^2}{2\sigma^2}\right).$$

This inequality establishes an upper bound on the probability that the estimation error for a specific feature basis g exceeds the margin ϵ.

Then, we aim to bound the maximum deviation across all $g \in \{1, \ldots, G\}$ by ϵ. The event where the maximum deviation exceeds ϵ is mathematically equivalent to the event that at least one component g has a deviation exceeding ϵ.

Applying Boole's inequality (the union bound), i.e., $\mathbb{P}(\cup A_i) \le \sum \mathbb{P}(A_i)$, we can upper bound the total failure probability

$$\mathbb{P}\Big(\max_{1\le g\le G}\big|\widehat{m}_g - m_g\big| > \epsilon\Big) = \mathbb{P}\left(\bigcup_{g=1}^{G}\Big\{\big|\widehat{m}_g - m_g\big| > \epsilon\Big\}\right) \le \sum_{g=1}^{G}\mathbb{P}\Big(\big|\widehat{m}_g - m_g\big| > \epsilon\Big).$$

Because the failure probability bound is identical for each g, substituting and summing over the G components gives

$$\mathbb{P}\Big(\max_{1\le g\le G}\big|\widehat{m}_g - m_g\big| > \epsilon\Big) \le \sum_{g=1}^{G} 2\exp\left(-\frac{B\epsilon^2}{2\sigma^2}\right) = 2G\exp\left(-\frac{B\epsilon^2}{2\sigma^2}\right).$$

To guarantee that the probability of the maximum error exceeding ϵ (the failure probability) is controlled below our predefined tolerance level δ, we enforce the condition

$$2G\exp\left(-\frac{B\epsilon^2}{2\sigma^2}\right) \le \delta.$$

Dividing by $2G$ and solving for B, we obtain

$$B \ge \frac{2\sigma^2}{\epsilon^2}\log\left(\frac{2G}{\delta}\right).$$

We can decompose the logarithmic term as $\log(2G/\delta) = \log(G/\delta) + \log(2)$. In an asymptotic sense (denoted by the Big-O or $\gtrsim$ notation), the absolute constant factor 2 can be absorbed. Therefore, provided that the batch size B satisfies:

$$B \gtrsim \frac{\sigma^2}{\epsilon^2}\log\left(\frac{G}{\delta}\right)$$

(where $\gtrsim$ hides absolute constants, such as 2 in this derivation), the probability that the maximum error exceeds ϵ is at most δ.

Taking the complement of this event (the success event) yields the final uniform bound

$$\mathbb{P}\Big(\max_{1\le g\le G}\big|\widehat{m}_g - \mathbb{E}[h\Psi_g]\big| \le \epsilon\Big) \ge 1-\delta.$$

□

Remark B.6 (When the assumption fails). The sub-Gaussian (in particular, uniformly bounded) assumption on $h\Psi_g$ is natural for polynomial dependence on bounded or Gaussian inputs, but it is violated by log-normal coefficient fields, whose exponential dependence on $\mathbf{Z}$ produces heavy tails and places substantial energy in high-degree modes. This is consistent with the reduced variance accuracy observed on the log-normal Darcy benchmark, and it can be detected before training with the subspace energy ratio (8).

B.5 Feature Gram/Orthogonality conditioning and its concentration

Let $F_{\mathrm{s}} \in \mathbb{R}^{B\times M}$ be the spatial feature matrix with rows $F_{\mathrm{s}}(b,:) = \big(\Phi_i(\mathbf{x}_b)\big)_{i=1}^{M}$ sampled from μ_x. Define the empirical Gram $\widehat{G}_\Phi = \frac{1}{B}F_{\mathrm{s}}^\top F_{\mathrm{s}}$ and population Gram $G_\Phi = \mathbb{E}_{\mathbf{x}\sim\mu_x}[\Phi(\mathbf{x})\Phi(\mathbf{x})^\top]$, where $\Phi(\mathbf{x}) = (\Phi_i(\mathbf{x}))_{i=1}^{M}$.

Theorem B.7 (Near-orthogonality ⇒ bounded condition number)**.** *If $\|\widehat{G}_\Phi - I\|_2 \le \delta < 1$, then the singular values of $F_{\mathrm{s}}/\sqrt{B}$ lie in $[\sqrt{1-\delta}, \sqrt{1+\delta}]$ and $\kappa(F_{\mathrm{s}}) \le \sqrt{\frac{1+\delta}{1-\delta}}$.*

Proof. Define the matrix $A = \frac{1}{\sqrt{B}}F_{\mathrm{s}}$. By the properties of the singular value decomposition (SVD), the squared singular values of A, denoted by $\sigma_j^2(A)$, coincide with the eigenvalues of the symmetric matrix $A^\top A$, denoted by $\lambda_j(A^\top A)$. Observing the expansion of $A^\top A$,

$$A^\top A = \left(\frac{1}{\sqrt{B}}F_{\mathrm{s}}\right)^\top\left(\frac{1}{\sqrt{B}}F_{\mathrm{s}}\right) = \frac{1}{B}F_{\mathrm{s}}^\top F_{\mathrm{s}}.$$

According to the given definition, this is precisely the empirical Gram matrix $\widehat{G}_\Phi$. Consequently, we establish

$$\lambda_j(\widehat{G}_\Phi) = \sigma_j^2\left(\frac{1}{\sqrt{B}}F_{\mathrm{s}}\right).$$

We are given the condition $\|\widehat{G}_\Phi - I\|_2 \leq \delta < 1$. For any real symmetric matrix M, its spectral norm (or 2-norm) is equal to the maximum absolute value of its eigenvalues, which implies

$$\|M\|_2 = \max_j |\lambda_j(M)|.$$

Substituting $M = \widehat{G}_\Phi - I$ yields that for all j, $|\lambda_j(\widehat{G}_\Phi - I)| \leq \delta$. Since the identity matrix I has all eigenvalues equal to 1 and shares the same eigenspace with $\widehat{G}_\Phi$, the eigenvalues of $\widehat{G}_\Phi - I$ can be expressed simply as $\lambda_j(\widehat{G}_\Phi) - 1$. Substituting this into the inequality gives

$$|\lambda_j(\widehat{G}_\Phi) - 1| \leq \delta.$$

Expanding the absolute value, we obtain the upper and lower bounds for all eigenvalues of the empirical Gram matrix

$$1 - \delta \leq \lambda_j(\widehat{G}_\Phi) \leq 1 + \delta,$$

which means

$$1 - \delta \leq \sigma_j^2 \left(\frac{1}{\sqrt{B}} F_{\mathrm{s}} \right) \leq 1 + \delta.$$

Since singular values are inherently non-negative ($\sigma_j \geq 0$) and the condition $\delta < 1$ ensures that $1 - \delta > 0$, we get

$$\sqrt{1 - \delta} \leq \sigma_j \left(\frac{1}{\sqrt{B}} F_{\mathrm{s}} \right) \leq \sqrt{1 + \delta}.$$

This concludes the proof of the first part: all singular values of $F_{\mathrm{s}}/\sqrt{B}$ are bounded within the interval $[\sqrt{1-\delta}, \sqrt{1+\delta}]$.

The spectral condition number (2-norm condition number) of a matrix X is defined as the ratio of its maximum singular value to its minimum singular value

$$\kappa(X) = \frac{\sigma_{\max}(X)}{\sigma_{\min}(X)}.$$

Importantly, scaling a matrix by a non-zero scalar does not alter its condition number, as both the maximum and minimum singular values are scaled by the exact same proportion. Thus,

$$\kappa(F_{\mathrm{s}}) = \kappa \left(\frac{1}{\sqrt{B}} F_{\mathrm{s}} \right).$$

Based on the singular value bounds, we have:

- Upper bound for the maximum singular value: $\sigma_{\max} \left(\frac{1}{\sqrt{B}} F_{\mathrm{s}} \right) \leq \sqrt{1 + \delta}$
- Lower bound for the minimum singular value: $\sigma_{\min} \left(\frac{1}{\sqrt{B}} F_{\mathrm{s}} \right) \geq \sqrt{1 - \delta}$

Substituting these bounds into the definition of the condition number yields the desired upper bound

$$\kappa(F_{\mathrm{s}}) \leq \frac{\sqrt{1+\delta}}{\sqrt{1-\delta}} = \sqrt{\frac{1+\delta}{1-\delta}}.$$

□

Theorem B.8 (Concentration of the sample Gram)**.** *Assume $\|\Phi(\mathbf{x})\|_2 \leq L$ almost surely. Then for any $\delta \in (0, 1)$, with probability $\geq 1 - \delta$.*

$$\left\| \widehat{G}_\Phi - G_\Phi \right\|_2 \lesssim L^2 \left(\sqrt{\tfrac{\log(M/\delta)}{B}} + \tfrac{\log(M/\delta)}{B} \right)$$

In particular, if $G_\Phi = I$ (population-orthonormalized features), then $\widehat{G}_\Phi$ is near-identity with the above rate.

Proof. Let $\Phi_b := \Phi(\mathbf{x}_b) \in \mathbb{R}^M$ be the spatial feature vector at the b-th collocation point, and define the centered, scaled rank-one summands

$$X_b \;:=\; \tfrac{1}{B} \big(\Phi_b \Phi_b^\top - G_\Phi \big), \qquad b = 1, \ldots, B.$$

By construction $\widehat{G}_\Phi - G_\Phi = \sum_{b=1}^{B} X_b$ and $\mathbb{E}[X_b] = 0$, so the deviation $\|\widehat{G}_\Phi - G_\Phi\|_2$ is a sum of B independent, mean-zero, symmetric matrices. The result therefore follows from the matrix Bernstein inequality once we control (i) a uniform almost-sure norm bound $R := \max_b \|X_b\|_2$ and (ii) a matrix-variance parameter $\sigma^2 := \big\| \sum_b \mathbb{E}[X_b^2] \big\|_2$. We compute these two ingredients next.

By the triangle inequality for matrix norms, we obtain

$$\|X_b\|_2 \le \frac{1}{B}\big(\|\Phi_b\Phi_b^\top\|_2 + \|G_\Phi\|_2\big).$$

We are given the condition $\|\Phi(\mathbf{x})\|_2 \le L$ almost surely (a.s.). Since $\Phi_b\Phi_b^\top$ is a rank-one matrix, its unique non-zero eigenvalue (which coincides with its spectral norm) is given by its trace

$$\|\Phi_b\Phi_b^\top\|_2 = \mathrm{Tr}(\Phi_b\Phi_b^\top) = \|\Phi_b\|_2^2 \le L^2.$$

By Jensen's inequality and the convexity of matrix norms, the spectral norm of the population covariance matrix is also bounded by

$$\|G_\Phi\|_2 = \|\mathbb{E}[\Phi_b\Phi_b^\top]\|_2 \le \mathbb{E}\big[\|\Phi_b\Phi_b^\top\|_2\big] \le L^2.$$

Substituting these two terms back yields a uniform upper bound on the norm

$$\|X_b\|_2 \le \frac{1}{B}(L^2 + L^2) = \frac{2L^2}{B} := R.$$

Consider the squared term $X_b^2 = \frac{1}{B^2}\big(\Phi_b\Phi_b^\top - G_\Phi\big)^2$. In terms of the positive semi-definite (PSD) partial ordering ($\preceq$), the second moment of a centered random matrix is always bounded above by the second moment of the uncentered matrix (i.e., $\mathbb{E}[(Y - \mathbb{E}Y)^2] \preceq \mathbb{E}[Y^2]$). Then, we have

$$\mathbb{E}[X_b^2] \preceq \frac{1}{B^2}\mathbb{E}\big[(\Phi_b\Phi_b^\top)^2\big].$$

Expanding the squared term, we have

$$(\Phi_b\Phi_b^\top)^2 = (\Phi_b\Phi_b^\top)(\Phi_b\Phi_b^\top) = \Phi_b(\Phi_b^\top\Phi_b)\Phi_b^\top = \|\Phi_b\|_2^2(\Phi_b\Phi_b^\top).$$

Since $\|\Phi_b\|_2^2 \le L^2$, it follows that

$$(\Phi_b\Phi_b^\top)^2 \preceq L^2(\Phi_b\Phi_b^\top).$$

Taking the expectation and summing over all $b = 1, \ldots, B$, we bound the total variance matrix as

$$\sum_{b=1}^{B} \mathbb{E}[X_b^2] \preceq \sum_{b=1}^{B} \frac{L^2}{B^2}\mathbb{E}[\Phi_b\Phi_b^\top] = \frac{L^2}{B}G_\Phi.$$

Evaluating the spectral norm yields the variance parameter σ^2,

$$\sigma^2 = \Big\| \sum_{b=1}^{B} \mathbb{E}[X_b^2] \Big\|_2 \le \frac{L^2}{B}\|G_\Phi\|_2 \le \frac{L^4}{B}.$$

The Matrix Bernstein inequality states that for any $t \ge 0$, we have

$$\mathbb{P}\left(\left\|\sum_{b=1}^{B} X_b\right\|_2 \ge t\right) \le 2M \exp\left(-\frac{t^2/2}{\sigma^2 + Rt/3}\right).$$

Setting the failure probability bound equal to the given confidence level δ, we obtain

$$2M \exp\left(-\frac{t^2/2}{\sigma^2 + Rt/3}\right) = \delta \implies \frac{t^2/2}{\sigma^2 + Rt/3} = \log\left(\frac{2M}{\delta}\right).$$

Solving this quadratic equation for t yields the high-probability upper bound for t given by

$$t \le \sqrt{2\sigma^2 \log\left(\frac{2M}{\delta}\right)} + \frac{2}{3}R\log\left(\frac{2M}{\delta}\right).$$

Substituting the bounds, specifically $\sigma^2 \le \frac{L^4}{B}$ and $R = \frac{2L^2}{B}$, we obtain

$$t \le \sqrt{\frac{2L^4}{B} \log\left(\frac{2M}{\delta}\right)} + \frac{4L^2}{3B} \log\left(\frac{2M}{\delta}\right).$$

Factoring out L^2 and subsuming the absolute constants (e.g., $\sqrt{2}$, $4/3$, and the constant offset in the logarithmic term, treating $\log(2M/\delta) \approx \log(M/\delta)$) into the asymptotic $\lesssim$ notation, we arrive at

$$t \lesssim L^2 \left(\sqrt{\frac{\log(M/\delta)}{B}} + \frac{\log(M/\delta)}{B}\right).$$

This implies that with probability at least $1 - \delta$, the deviation is bounded by

$$\|\widehat{G}_\Phi - G_\Phi\|_2 \lesssim L^2 \left(\sqrt{\frac{\log(M/\delta)}{B}} + \frac{\log(M/\delta)}{B}\right).$$

Notably, when $G_\Phi = I$, this result immediately demonstrates that the empirical Gram matrix $\widehat{G}_\Phi$ approaches the identity matrix at the derived rate (satisfying the near-identity property), which asymptotically guarantees the approximate orthogonality of the features. □

B.6 CP kernel: algebra, efficiency, and gauge

Consider the rank-q CP parameterization $C_{ikg} = \sum_{r=1}^{q} A_{ir} B_{kr} D_{gr}$ and the batched forward $u = ((F_{\mathrm{s}}A) \odot (F_{\mathrm{t}}B) \odot (F_{\mathrm{z}}D))\, w$.

Theorem B.9 (Parameter and arithmetic complexity)**.** *The dense tensor parameterization uses $O(MM_tG)$ parameters and $O(BMM_tG)$ arithmetic per batch, whereas the CP kernel uses $O((M{+}M_t{+}G)q)$ parameters and $O(B\,q\,(M{+}M_t{+}G))$ arithmetic.*

Proof. Parameter count. The dense parameterization stores every entry C_{ikg} of the 3-way coupling tensor independently, so it uses MM_tG scalar parameters, i.e. $O(MM_tG)$. The CP kernel instead stores three factor matrices of shapes $M{\times}q$, $M_t{\times}q$, $G{\times}q$ and a weight vector $w \in \mathbb{R}^q$, totalling $q(M + M_t + G) + q = O(q(M + M_t + G))$ scalar parameters.

Arithmetic per batch. The dense forward $u_b = \sum_{i,k,g} C_{ikg} \Phi_i(\mathbf{x}_b) T_k(t_b) \Psi_g(\mathbf{Z}_b)$ touches every entry of C for every sample, costing $\Theta(MM_tG)$ FLOPs per sample and $O(BMM_tG)$ over a batch of size B. The CP forward $u = ((F_{\mathrm{s}}A) \odot (F_{\mathrm{t}}B) \odot (F_{\mathrm{z}}D))w$ first multiplies the three feature matrices $F_{\mathrm{s}} \in \mathbb{R}^{B\times M}$, $F_{\mathrm{t}} \in \mathbb{R}^{B\times M_t}$, $F_{\mathrm{z}} \in \mathbb{R}^{B\times G}$ by the corresponding factors at cost $O(BqM) + O(BqM_t) + O(BqG) = O(Bq(M + M_t + G))$, then performs an $O(Bq)$ Hadamard fusion and an $O(Bq)$ inner product with w, for an overall cost $O(Bq(M + M_t + G))$. Hence the CP kernel reduces both parameter count and per-batch arithmetic from multiplicative $O(MM_tG)$ to additive $O(q(M{+}M_t{+}G))$, an asymptotic improvement whenever $q \ll MM_tG/(M{+}M_t{+}G)$. □

Theorem B.10 (Gauge invariance and normalization)**.** *For any invertible diagonal $\Lambda_1, \Lambda_2, \Lambda_3 \in \mathbb{R}^{q\times q}$, the mapping*

$$(A, B, D, w) \;\mapsto\; \big(A\Lambda_1,\; B\Lambda_2,\; D\Lambda_3\,\Lambda_1^{-1}\Lambda_2^{-1}\Lambda_3^{-1} w\big)$$

leaves u unchanged. Column-wise normalization of (A, B, D) fixes a gauge (up to column permutations/signs) and improves Lipschitz conditioning of the forward map with respect to each factor.

Proof. Let $\Lambda_1, \Lambda_2, \Lambda_3 \in \mathbb{R}^{q\times q}$ be invertible diagonal matrices. We define the transformed parameters as follows

$$\tilde{A} = A\Lambda_1, \tag{9}$$

$$\tilde{B} = B\Lambda_2, \tag{10}$$

$$\tilde{D} = D\Lambda_3, \tag{11}$$

$$\tilde{w} = \Lambda_1^{-1}\Lambda_2^{-1}\Lambda_3^{-1} w. \tag{12}$$

The computation for the forward propagation is formulated as

$$\tilde{u} = \big((F_{\mathrm{s}}\tilde{A}) \odot (F_{\mathrm{t}}\tilde{B}) \odot (F_{\mathrm{z}}\tilde{D})\big)\tilde{w}.$$

First, we consider the transformation of the feature projection matrices. Since F_{s} is post-multiplied by $\tilde{A}$ and Λ_1 is a diagonal matrix, this operation is equivalent to scaling the columns of $F_{\mathrm{s}}A$, which means

$$F_{\mathrm{s}}\tilde{A} = F_{\mathrm{s}}(A\Lambda_1) = (F_{\mathrm{s}}A)\Lambda_1.$$

Similarly, we have

$$F_{\mathrm{t}}\tilde{B} = (F_{\mathrm{t}}B)\Lambda_2, \tag{13}$$

$$F_{\mathrm{z}}\tilde{D} = (F_{\mathrm{z}}D)\Lambda_3. \tag{14}$$

Next, we examine the Hadamard product (denoted by $\odot$). For arbitrary matrices X, Y and diagonal matrices Λ_X, Λ_Y, their Hadamard product satisfies

$$(X\Lambda_X) \odot (Y\Lambda_Y) = (X \odot Y)(\Lambda_X\Lambda_Y).$$

This holds because post-multiplying by a diagonal matrix essentially scales the columns of the target matrix, and the scaling factors for the r-th column are directly multiplied together. Expanding the Hadamard product of the three transformed matrices, we obtain

$$\big((F_{\mathrm{s}}A)\Lambda_1\big) \odot \big((F_{\mathrm{t}}B)\Lambda_2\big) \odot \big((F_{\mathrm{z}}D)\Lambda_3\big) = \big((F_{\mathrm{s}}A) \odot (F_{\mathrm{t}}B) \odot (F_{\mathrm{z}}D)\big)(\Lambda_1\Lambda_2\Lambda_3).$$

Finally, we multiply this result by the transformed weight vector $\tilde{w}$ and we get

$$\tilde{u} = \big((F_{\mathrm{s}}A) \odot (F_{\mathrm{t}}B) \odot (F_{\mathrm{z}}D)\big)(\Lambda_1\Lambda_2\Lambda_3)\big(\Lambda_1^{-1}\Lambda_2^{-1}\Lambda_3^{-1}w\big).$$

Since multiplication between diagonal matrices is commutative, and $\Lambda_i\Lambda_i^{-1} = I$ (the identity matrix), all the diagonal matrices cancel out, yielding

$$(\Lambda_1\Lambda_2\Lambda_3)(\Lambda_1^{-1}\Lambda_2^{-1}\Lambda_3^{-1}) = I.$$

Therefore,

$$\tilde{u} = \big((F_{\mathrm{s}}A) \odot (F_{\mathrm{t}}B) \odot (F_{\mathrm{z}}D)\big)w = u.$$

This concludes the proof that the forward propagation mapping is strictly gauge invariant under the aforementioned transformations.

As demonstrated in the preceding proof, for any non-zero constant λ, we can scale the r-th column of the factor matrix A by λ while simultaneously dividing the r-th element of the weight w by λ without altering the network output. This symmetry induces infinitely many equivalent global optima in the parameter space (i.e., continuous scaling valleys).

To eliminate this continuous ambiguity, we apply column normalization to A, B, and D (typically by constraining the L_2 norm of each column to 1). Enforcing the constraint $\|A_{:,r}\|_2 = 1$ strictly limits the permissible diagonal scaling factors to $|\lambda| = 1$. This implies that the diagonal entries of Λ_1, Λ_2, and Λ_3 are restricted to ± 1. Consequently, the continuous scaling ambiguity collapses into discrete sign permutations. Combined with the inherent column permutation symmetry of the rank-1 components in CP decomposition, we successfully fix the gauge up to permutations and signs.

In neural network optimization, the smoothness of the loss function with respect to a specific parameter (e.g., matrix A) is governed by the Lipschitz constant of its partial derivative. In the multilinear CP format, the partial derivative of u with respect to A heavily depends on the magnitudes of B, D, and w. Roughly speaking, the gradient norm is proportional to

$$\left\| \frac{\partial u}{\partial A} \right\| \propto \|B\| \cdot \|D\| \cdot \|w\|.$$

Without normalization constraints, the optimizer might drift into ill-conditioned regions where certain columns of B and D become pathologically large (e.g., 10^6) while w becomes vanishingly small (e.g., 10^{-12}) to compensate. Although this leaves u invariant, it causes the gradient with respect to A to explode, driving the Lipschitz constant toward infinity and severely destabilizing the optimization process (Gradient Explosion).

By enforcing column normalization on A, B, and D, their norms are rigidly bounded near 1. The network is thus compelled to absorb all global magnitude information into the unique linear weight vector w. This provides well-bounded control over the Jacobians of the factor matrices, significantly improving the Lipschitz conditioning of the forward mapping with respect to each factor. As a result, the optimization process (such as gradient descent) becomes substantially more stable and efficient. □

B.7 Closed-form derivatives for Gaussian atoms and stability under warps

For $\varphi(\mathbf{x}) = \exp\big(-\frac{1}{2}(\mathbf{x}-\mu)^\top \Sigma^{-1}(\mathbf{x}-\mu)\big)$ with $\Sigma \succ 0$,

$$\nabla\varphi(\mathbf{x}) = -\Sigma^{-1}(\mathbf{x}-\mu)\,\varphi(\mathbf{x}),$$
$$\Delta\varphi(\mathbf{x}) = \big((\mathbf{x}-\mu)^\top \Sigma^{-2}(\mathbf{x}-\mu) - \operatorname{tr}\Sigma^{-1}\big)\,\varphi(\mathbf{x}),$$

and for a $\mathcal{C}^1$ residual warp $\mathbf{W}(\mathbf{x}) = \mathbf{x} + \mathbf{w}_\theta(\mathbf{x})$, derivatives of $\varphi \circ \mathbf{W}$ follow by the chain rule with Jacobian $J_{\mathbf{W}}(\mathbf{x})$. If $\|J_{\mathbf{W}} - I\|_2 \le \varepsilon < 1$, the mapping $\Phi_i \mapsto \Phi_i \circ \mathbf{W}$ is bi-Lipschitz on $L^2(\Omega)$ with constants depending only on ε and Ω.

The three statements above are direct calculus consequences and are recorded here for completeness.

(i) Gradient and Laplacian of a Gaussian atom. Write $\varphi(\mathbf{x}) = \exp(-\frac{1}{2}q(\mathbf{x}))$ with $q(\mathbf{x}) = (\mathbf{x}-\mu)^\top\Sigma^{-1}(\mathbf{x}-\mu)$. Since Σ is symmetric, $\nabla q(\mathbf{x}) = 2\Sigma^{-1}(\mathbf{x}-\mu)$, and the chain rule gives

$$\nabla\varphi(\mathbf{x}) = -\tfrac{1}{2}\,\nabla q(\mathbf{x})\,\varphi(\mathbf{x}) = -\Sigma^{-1}(\mathbf{x}-\mu)\,\varphi(\mathbf{x}).$$

Taking divergences and using the product rule $\nabla\cdot(f\mathbf{v}) = \mathbf{v}^\top\nabla f + f\,\nabla\cdot\mathbf{v}$ with $\mathbf{v} = \Sigma^{-1}(\mathbf{x}-\mu)$ (so $\nabla\cdot\mathbf{v} = \operatorname{tr}\Sigma^{-1}$),

$$\Delta\varphi(\mathbf{x}) = \nabla\cdot(\nabla\varphi) = \big((\mathbf{x}-\mu)^\top\Sigma^{-2}(\mathbf{x}-\mu) - \operatorname{tr}\Sigma^{-1}\big)\varphi(\mathbf{x}),$$

matching the displayed formula.

(ii) Composition with a warp. For any $\mathcal{C}^1$ map $\mathbf{W}$, the multivariate chain rule gives $\nabla(\varphi\circ\mathbf{W})(\mathbf{x}) = J_{\mathbf{W}}(\mathbf{x})^\top(\nabla\varphi)(\mathbf{W}(\mathbf{x}))$, and the Laplacian follows from $\Delta(\varphi\circ\mathbf{W}) = \operatorname{tr}\big(J_{\mathbf{W}}^\top(\operatorname{Hess}\varphi)(\mathbf{W})\,J_{\mathbf{W}}\big) + (\nabla\varphi)(\mathbf{W})^\top\,\Delta\mathbf{W}$. For a residual warp $\mathbf{W}(\mathbf{x}) = \mathbf{x} + \mathbf{w}_\theta(\mathbf{x})$, $J_{\mathbf{W}} = I + J_{\mathbf{w}_\theta}$ and all quantities are computable in closed form once $\mathbf{w}_\theta$ and its first/second derivatives are available from the warp MLP.

(iii) Bi-Lipschitz stability under small Jacobian perturbation. Suppose $\|J_{\mathbf{W}}(\mathbf{x}) - I\|_2 \le \varepsilon < 1$ uniformly on Ω. By the Neumann series, $J_{\mathbf{W}}$ is invertible everywhere with $\|J_{\mathbf{W}}^{-1}\|_2 \le 1/(1-\varepsilon)$ and $\|J_{\mathbf{W}}\|_2 \le 1+\varepsilon$. The mean-value theorem on the convex domain Ω then yields the two-sided bound

$$(1-\varepsilon)\,\|\mathbf{x}-\mathbf{x}'\|_2 \;\le\; \|\mathbf{W}(\mathbf{x}) - \mathbf{W}(\mathbf{x}')\|_2 \;\le\; (1+\varepsilon)\,\|\mathbf{x}-\mathbf{x}'\|_2,$$

so $\mathbf{W}$ is bi-Lipschitz with constants $(1\pm\varepsilon)$, and a standard change-of-variables argument shows that $\Phi_i \mapsto \Phi_i\circ\mathbf{W}$ is a bi-Lipschitz map on $L^2(\Omega)$ with constants depending only on ε (through the Jacobian determinant $\det J_{\mathbf{W}} \in [(1-\varepsilon)^d, (1+\varepsilon)^d]$) and Ω. In particular, near-orthogonality of the unwarped dictionary is preserved up to a factor controlled by ε, which is why the warp does not destabilize the Gram conditioning.

C Manufactured Solution Definitions

All four benchmark problems are defined on the spatio-temporal domain $\Omega = [-1,1]^2$, $\mathcal{T} = [0,1]$, with R i.i.d. standard-normal random variables $\mathbf{Z} = (Z_1,\dots,Z_R) \sim \mathcal{N}(0, I_R)$. These four equations are standard benchmarks in the high-dimensional PDE literature Fan et al. [2026].

Karhunen–Loève coefficient. All problems share a log-normal random field:

$$k(\mathbf{x},\mathbf{Z}) = \exp\Big(\mu_k + \sigma\sum_{i=1}^{R}\sqrt{\lambda_i}\,Z_i\,\phi_i(\mathbf{x})\Big),$$

where $\{\lambda_i, \phi_i\}$ are the eigenvalue–eigenfunction pairs of an exponential covariance kernel $C(\mathbf{x},\mathbf{x}') = \exp(-\|\mathbf{x}-\mathbf{x}'\|_1/\ell)$ with correlation length $\ell = 1$, $\mu_k = 0$, and $\sigma = 1$.

1. Stochastic Diffusion.

$$\frac{\partial u}{\partial t} - \nabla\cdot\big(k(\mathbf{x},\mathbf{Z})\,\nabla u\big) = s(\mathbf{x},t,\mathbf{Z}),$$

with manufactured exact solution

$$u^*(\mathbf{x},t,\mathbf{Z}) = \sin(\pi x)\,\sin(\pi y)\,e^{-t}\,\Big(1+0.1\textstyle\sum_{i=1}^{R} Z_i\Big),$$

where the source s is obtained by substituting u^* into the PDE. Dirichlet boundary conditions $u\big|_{\partial\Omega} = 0$ and initial condition $u(0,\mathbf{x},\mathbf{Z}) = u^*(0,\mathbf{x},\mathbf{Z})$ are imposed.

2. Stochastic Allen–Cahn.

$$\frac{\partial u}{\partial t} - \nabla\cdot\big(k\,\nabla u\big) - (u-u^3) = s,$$

with the *same* manufactured solution as diffusion. The cubic reaction term $u - u^3$ introduces additional nonlinearity.

3. Stochastic Burgers.

$$\frac{\partial u}{\partial t} + u\left(\frac{\partial u}{\partial x} + \frac{\partial u}{\partial y}\right) - \nu\,\Delta u = s,$$

with kinematic viscosity $\nu = 0.01$ and the same manufactured solution as diffusion. The convective nonlinearity $u\,\nabla u$ makes this the most challenging benchmark for standard PINNs.

4. Stochastic Darcy (divergence form).

$$\frac{\partial u}{\partial t} - \nabla\cdot\big(k\,\nabla u\big) = s,$$

with a different manufactured solution that couples the solution to the stochastic field more tightly:

$$u^*(\mathbf{x},t,\mathbf{Z}) = \sin(\pi x)\sin(\pi y)\,e^{-t}\exp\Big(\gamma\sum_{i=1}^{R}\sqrt{\lambda_i}\,Z_i\,\phi_i(\mathbf{x})\Big),$$

where $\gamma = 0.3$. Unlike the linear Z-dependence in the first three problems, the Darcy solution depends exponentially on $\mathbf{Z}$ through the KL expansion, requiring higher gPC orders for accurate recovery.

5. Non-manufactured Stochastic Poisson (forward simulation). We solve a stochastic Poisson equation with a random source term:

$$-\Delta u = f(x,y,\mathbf{Z}), \quad \text{with } u = 0 \text{ on } \partial\Omega,$$

where $f(x,y,\mathbf{Z}) = 1 + \sigma\sum_{i=1}^{R}\sqrt{\lambda_i}\,Z_i\,\phi_i(\mathbf{x})$ is a KL-expanded random source field (σ=0.5, R=5). The reference solution is computed numerically via finite differences on a 64×64 grid over 10,000 Monte Carlo samples.

6. Non-manufactured Stochastic Darcy (forward simulation). We also solve a stochastic Darcy flow with a log-normal permeability field:

$$-\nabla\cdot\big(k(x,y,\mathbf{Z})\,\nabla u\big) = 1.0, \quad \text{with } u = 0 \text{ on } \partial\Omega,$$

where $k(x,y,\mathbf{Z}) = \exp\big(\sigma\sum_{i=1}^{R}\sqrt{\lambda_i}\,Z_i\,\phi_i(\mathbf{x})\big)$ with σ=1.0 and R=5. The physical response $u(x,y,\mathbf{Z})$ is purely driven by the stochastic permeability field k and the constant source term. The reference solution is computed via the same finite-difference Monte Carlo procedure (N=10,000, 64×64 grid).

D Details of Baseline Methods

In this section, we record the exact baseline protocols used in the main experiments. The primary comparisons include six neural or classical baselines (Vanilla PINN, Vanilla PINN+proj, PI-DeepONet, MC-Dropout PINN, Deep Ensembles, stochastic collocation, SPINN+Z, functional CP-PINN, and Neural Chaos), and PC^2 is another physics-informed polynomial-chaos baseline for the selected forward benchmarks.

Common training configuration (all neural baselines). Unless otherwise noted, every neural baseline shares the following implementation choices, which match those of S^2-PINN to ensure a fair comparison:

- *Optimizer.* Adam with $(\beta_1, \beta_2) = (0.9, 0.999)$, $\epsilon = 10^{-8}$, no learning-rate scheduler (except for the tuned PI-DeepONet, see below), and no gradient clipping. The base learning rate is 10^{-3} for diffusion / Allen–Cahn / Burgers and 5×10^{-4} for Darcy.
- *Initialization.* All `nn.Linear` layers use Xavier-normal weights and zero biases.
- *Loss weights.* For diffusion / Allen–Cahn / Burgers, $w_{\mathrm{pde}} = w_{\mathrm{bc}} = w_{\mathrm{ic}} = 1.0$; for Darcy, $w_{\mathrm{bc}} = w_{\mathrm{ic}} = 10.0$ to compensate for the steeper boundary layers. The projection weight w_{proj} is 1.0 for "+proj" variants and 0 otherwise (see per-baseline notes below).
- *Training horizon.* 20,000 Adam steps for diffusion / Allen–Cahn / Burgers and 30,000 for Darcy.
- *Sampling strategy.* All collocation, boundary, and initial-condition batches are resampled *every* step (`resample_every` $=1$). Collocation points are drawn uniformly at random in $\mathcal{T}\times\Omega$; boundary points draw $t \sim \mathcal{U}(\mathcal{T})$, the free spatial coordinate $\sim \mathcal{U}([-1,1])$, and the boundary edge index $\sim \mathrm{Uniform}\{1,2,3,4\}$; initial-condition points fix $t = t_{\min}$ with uniform (x, y). Random inputs are drawn from $\mathcal{N}(0, I_R)$ for Gaussian configurations and from $\mathcal{U}([-1,1]^R)$ for uniform configurations.

Vanilla PINN. The standard physics-informed neural network represents the solution $u(t, \mathbf{x}, \mathbf{Z})$ as a fully connected Multi-Layer Perceptron (MLP) taking the concatenated space, time, and stochastic variables as inputs. We use a 4-layer MLP with 256 hidden units per layer and hyperbolic tangent ($\tanh$) activations. Concretely, the architecture is $(3{+}R) \to 256 \to 256 \to 256 \to 256 \to 1$ with $\tanh$ on every hidden layer and a linear output; for the main stochastic runs $R{=}5$, giving an input dimension of 8. The loss function is defined as

$$\mathcal{L}_{\text{PINN}} = w_{\text{pde}}\mathbb{E}[r^2] + w_{\text{bc}}\mathbb{E}_{\partial\Omega}[(u-u^*)^2] + w_{\text{ic}}\mathbb{E}[(u-u^*)^2|_{t=0}]. \tag{15}$$

The network relies purely on dense sampling and gradient descent to learn the complex high-dimensional stochastic dependence without any explicit polynomial basis.

Vanilla PINN + proj. This baseline uses the exact same MLP architecture as the Vanilla PINN but augments the standard loss function with the gPC Galerkin projection term introduced in our method: $\mathcal{L}_{\text{PINN+proj}} = \mathcal{L}_{\text{PINN}} + w_{\text{proj}} \sum_{g=1}^{G} \left(\mathbb{E}[r\Psi_g(\mathbf{Z})]\right)^2$, with $w_{\mathrm{proj}}{=}1.0$ matching the S^2-PINN setting. This isolates the effect of the physics objective from the separable tensor architecture.

PI-DeepONet. Physics-Informed DeepONet Goswami et al. [2023] utilizes a dual-network architecture consisting of a Branch net and a Trunk net. The Branch net takes the stochastic parameters $\mathbf{Z}$ and outputs q latent coefficients. The Trunk net processes the spatiotemporal coordinates $(t, \mathbf{x})$ and outputs q basis functions. The final solution is computed as the inner product of these two latent vectors: $u(t, \mathbf{x}, \mathbf{Z}) = \sum_{k=1}^{q} b_k(\mathbf{Z})\tau_k(t, \mathbf{x}) + b_0$, where b_0 is a single learnable scalar bias added to the inner product. Concretely, both branches use 3 hidden layers of width 128 with $\tanh$ activations and a linear projection to a $q{=}128$-dimensional latent: Branch is $R \to 128 \to 128 \to 128 \to 128$ (so input dim 5 for $R{=}5$) and Trunk is $3 \to 128 \to 128 \to 128 \to 128$ (taking (t, x, y)). Weights are Xavier-normal initialised with zero biases. It is trained using the standard $\mathcal{L}_{\text{PINN}}$ loss. Because the default configuration diverged on Allen–Cahn, the reported results use a tuned setting: learning rate 10^{-3}, and on Allen–Cahn and Darcy a `ReduceLROnPlateau` scheduler with factor 0.5, patience 3, and minimum learning rate 10^{-6}.

Derivative-informed operator training. Rather than constraining the residual, derivative-informed objectives supervise the input–output sensitivities of a neural operator, which matters when the surrogate is differentiated inside inverse problems or PDE-constrained optimization. Yang et al. Yang et al. [2026] enforce this consistency entirely on-the-fly: a sketched tangent consistency loss derives the sensitivity constraint directly from the governing equation, so no offline tangent labels are required and the operator architecture is unchanged. We do not include it as a direct baseline because it targets tangent-level supervision of a *deterministic* operator and carries no explicit stochastic basis, so it does not produce calibrated moments for random PDEs.

MC-Dropout PINN. To provide a strong probabilistic baseline for uncertainty quantification, we augment the Vanilla PINN with Dropout layers (with drop probability $p = 0.1$) after every hidden layer. Concretely, a dropout layer is inserted after the activation of every hidden layer (but not after the linear output), so the architecture is $\big[\text{Linear} \to \tanh \to \text{Dropout}(p)\big]^4 \to \text{Linear}$ with the same Xavier-normal initialization as the Vanilla PINN. During training, dropout acts as a regularizer. During inference, dropout remains active, and we sample 50 forward passes for each input collocation point. This yields a predictive distribution that estimates the epistemic uncertainty of the network, which is then combined with the aleatoric variation over $\mathbf{Z}$ to compute coverage and calibration metrics. The training loss is $\mathcal{L}_{\text{PINN}}$.

Deep Ensembles. Following Lakshminarayanan et al. Lakshminarayanan et al. [2017], we train an ensemble of $K = 5$ independent Vanilla PINNs, each initialized with a different random seed. Each member uses the same Vanilla-PINN architecture (4 hidden layers $\times$ 256, $\tanh$) and the same 20,000-step training schedule, but draws its own collocation samples and weight initialization from the seed. Members are trained completely independently and then aggregated post-hoc: the prediction is the per-point mean and the epistemic uncertainty is the per-point standard deviation across the K members. Uncertainty-quantification metrics are computed with 1024 stochastic samples each for moment estimation, interval coverage, and held-out test points (`n_z_moment` $=$`n_z_interval` $=$ `n_z_test` $=1024$). Deep ensembles are widely considered a gold-standard baseline for predictive uncertainty. Unlike the other no-projection baselines, the Deep-Ensembles launch script does not override w_{proj}, so each member inherits w_{proj}=1.0 from the base configuration. This is intentional, since the strongest possible Deep-Ensembles baseline includes the same projection regularizer used by S^2-PINN.

Stochastic Collocation (SC). SC is a classical non-neural spectral method that leverages tensorized Gauss quadrature nodes in the stochastic space. For a quadrature order n_q, we define a grid of n_q^R nodes in $\mathbb{R}^R$. We then train a completely independent, deterministic spatial PINN $u_i(t, \mathbf{x})$ for each node $\mathbf{Z}_i$. The statistical moments of the random field are reconstructed post-training using the corresponding quadrature weights: $\bar{u}(t, \mathbf{x}) = \sum_i W_i\, u_i(t, \mathbf{x})$ and $\text{Var}(u)(t, \mathbf{x}) = \max\big(0,\, \sum_i W_i\, u_i(t, \mathbf{x})^2 - \bar{u}(t, \mathbf{x})^2\big)$, where $\{W_i\}$ are the tensor-product quadrature weights and the clamp at zero guards against round-off-induced negative variance. While highly accurate for smooth problems, the number of required PINNs scales exponentially with R, leading to severe computational bottlenecks (e.g., $n_q = 3$ for $R = 5$ requires 243 independent training runs). The reported tractable SC baseline uses n_q=2 per stochastic dimension, which gives 2^5=32 tensor-product nodes for R=5. Each node is a Vanilla-PINN-style spatial network trained for $5{,}000$ Adam steps on diffusion / Allen–Cahn / Burgers and $8{,}000$ steps on Darcy. Reference Monte-Carlo statistics for SC's relative-L^2 metrics are computed with $4{,}096$ stochastic samples and $2{,}048$ spatial probes ($n_{\text{mc,ref}}$=4096, n_{probes}=2048).

Physics-informed polynomial chaos (PC^2). We implement the PC^2 baseline of Novák et al. Novák et al. [2024] using UQpy's physics-informed PCE stack. Dirichlet data are used as boundary regression data, while PDE residual constraints are enforced at virtual collocation points through the KKT system in `ConstrainedPCE`. To avoid an unfair capacity mismatch, we do not use UQpy's native total-degree basis over all variables as the main comparison; instead, we use an anisotropic product basis with stochastic total degree fixed at p_Z=3 and deterministic degree p_{det} swept over $\{3, 4, 5\}$ for diffusion and $\{4, 6, 8\}$ for Poisson/Darcy. Model selection uses only held-out PDE/boundary residuals. This keeps the stochastic gPC capacity aligned with S^2-PINN while giving PC^2 a reasonable deterministic polynomial space.

SPINN+Z. A separable PINN Cho et al. [2023] in which the random variables $Z_1, \ldots, Z_R$ are treated as additional separable input coordinates, each processed by its own coordinate network, with the per-coordinate features combined by the SPINN outer-product/summation rule. It is trained with $\mathcal{L}_{\text{PINN}}$ and contains no explicit stochastic basis or projection term.

Functional CP-PINN. The functional tensor decomposition PINN of Vemuri et al. Vemuri et al. [2024], in which each coordinate (space, time, and random variables) is assigned a separate subnetwork and the outputs are coupled by a CP outer product. It is trained with $\mathcal{L}_{\text{PINN}}$.

Neural Chaos. Neural Chaos Bahmani et al. [2025] replaces the fixed polynomial chaos basis with sequentially learned neural basis functions. As a data-driven operator, it is trained on reference solution realizations and therefore acts as a supervised oracle rather than a purely physics-informed competitor; a physics residual is only evaluated post hoc. Since diffusion, Allen–Cahn, and Burgers share the same manufactured solution field, a single shared checkpoint is used for these three equations, and only the equation-specific post-hoc residual differs.

Architectural PINN variants. Two recent PINN architectures improve the base network rather than the stochastic treatment. compleX-PINN Si et al. [2026] replaces the activation with a learnable form derived from Cauchy's integral theorem, obtaining high accuracy with a single hidden layer and reporting gains on high-dimensional problems. IDPINN Si and Yan [2025] decomposes the domain into subdomains with dedicated networks and initializes training from a small-data pretrained model, improving interface smoothness and convergence speed. We do not include either as a baseline because both are *deterministic* architecture or optimization devices: they carry no explicit stochastic basis and therefore produce no calibrated moments or variance estimates for random PDEs. Their improvements are orthogonal to ours in principle and could be combined with the separable gPC representation introduced here.

E Model Architectures, Parameter Counts, and Training Configuration

Table 8 summarises the architecture and learnable parameter count for the main neural baselines and stochastic collocation. PC^2 is excluded from this parameter table because its capacity is better described by polynomial basis size rather than neural weights. S^2-PINN uses roughly $30\times$ fewer neural parameters than a single Vanilla PINN because the CP kernel couples three compact, purpose-built feature extractors instead of a monolithic MLP.

Table 8: Architecture details and parameter counts for all compared methods on the $R{=}5$ stochastic diffusion benchmark. M: spatial atoms; M_t: temporal features; G: gPC basis size; q: CP rank; h: hidden width; L: number of hidden layers.

Method	Architecture	Training steps	#Params
S^2-PINN	SpatialDict2D ($M{=}256$, warp MLP $2{\to}64{\to}2$) + TimeBasis ($K{=}8 \Rightarrow M_t{=}17$) + GPCBasis ($R{=}5, p{=}3 \Rightarrow G{=}56$) + CPCore ($q{=}16$)	20 000	7 018
Vanilla PINN	MLP $8{\to}256{\to}256{\to}256{\to}256{\to}1$, tanh	20 000	199 937
Vanilla PINN+proj	same + gPC projection loss	20 000	199 937
MC-Dropout PINN	same + dropout $p{=}0.1$	20 000	199 937
PI-DeepONet	Branch: $5{\to}128{\to}128{\to}128{\to}128$ Trunk: $3{\to}128{\to}128{\to}128{\to}128$	20 000	101 633
SC ($n_q{=}2$, 32 nodes)	$32\times$ Vanilla PINN	5 000 each	6 397 504

All experiments are based on NVIDIA T4. The S^2-PINN parameter count decomposes as

- *Spatial dictionary*: 256 centres ($\mu \in \mathbb{R}^{256\times 2}$), 256 log-bandwidths, warp MLP ($2{\times}64{+}64{+}64{\times}2{+}2 = 194$ params) $\Rightarrow$ 962.
- *Temporal basis*: 8 learnable log-frequencies $\Rightarrow$ 8.
- *gPC basis*: fixed Hermite polynomials (no learnable params).
- *CP core*: $A{\in}\mathbb{R}^{256\times 16}$, $B{\in}\mathbb{R}^{17\times 16}$, $D{\in}\mathbb{R}^{56\times 16}$, $w{\in}\mathbb{R}^{16} \Rightarrow$ 5 280.
- *Warp MLP*: $2{\times}64{+}64{+}64{\times}2{+}2 = 194 \Rightarrow$ included above. Biases account for the remainder up to 7 018.

Training algorithm and default hyperparameters. The pseudocode Algorithm 1 summarizes the training loop.

Algorithm 1 Training loop

Initialize dictionary/warp parameters, CP factors (A, B, D, w)
for $s = 1$ to S **do**
  if $s \bmod S_{\text{rs}} = 1$ **then**
    Resample PDE / BC / IC / Gram batches
  end if
  Build features $\mathbf{F}_{\text{space}}, \mathbf{F}_{\text{time}}, \mathbf{F}_{\text{stoc}}$
  Compute u, $\partial_t u$, and Δu (analytic time, autodiff space)
  Evaluate residuals r and all loss terms in (5)
  Adam step
  Column-normalize CP factors; update moving stats for Gram if used
end for

Default Hyperparameters. For the four benchmark problems in Tables 1–2, we use

$$M = 256, \quad M_t = 17, \quad (R, p) = (5, 3) \Rightarrow G = \tbinom{8}{3} = 56, \quad q = 16,$$

loss weights

$$w_{\text{pde}} = w_{\text{proj}} = w_{\text{bc}} = w_{\text{ic}} = w_{\text{orth}} = 1.0,\ \lambda_{\text{fac}} = 0.1,$$

gradient accumulation $\kappa = 1$, resampling period $S_{\text{rs}} = 1$, and collocation batch sizes of $32{,}768$ for PDE, boundary, and initial conditions.

F Ablation Study Details

We conducted controlled ablations on the stochastic diffusion benchmark to isolate the contribution of each hyperparameter. All experiments report errors at final time t=1.0. Tables 9–12 present the full results for baseline architecture, spatial atoms, gPC order, and CP rank, respectively.

Table 9: Baseline Architecture Ablation (hidden size × layers)

Configuration	$\mathcal{E}_{\bar{u}}$	$\mathcal{E}_{u_1}$	$\mathcal{E}_{u_2}$
default (baseline)	$(4.23 \pm 1.03) \times 10^{-3}$	$(3.97 \pm 1.39) \times 10^{-3}$	$(4.25 \pm 1.07) \times 10^{-3}$
$h = 128, L = 4$	$(4.41 \pm 0.63) \times 10^{-2}$	$(4.10 \pm 0.34) \times 10^{-2}$	$(3.86 \pm 0.64) \times 10^{-2}$
$h = 128, L = 6$	$(2.33 \pm 0.44) \times 10^{-2}$	$(2.12 \pm 0.44) \times 10^{-2}$	$(2.26 \pm 0.38) \times 10^{-2}$
$h = 256, L = 4$	$(3.26 \pm 1.39) \times 10^{-2}$	$(3.09 \pm 1.09) \times 10^{-2}$	$(3.04 \pm 1.01) \times 10^{-2}$
$h = 256, L = 6$	$(6.10 \pm 2.69) \times 10^{-2}$	$(5.81 \pm 2.56) \times 10^{-2}$	$(5.68 \pm 2.58) \times 10^{-2}$

Table 10: Spatial Atoms (M) Ablation

Configuration	$\mathcal{E}_{\bar{u}}$	$\mathcal{E}_{u_1}$	$\mathcal{E}_{u_2}$
default (baseline)	$(4.23 \pm 1.03) \times 10^{-3}$	$(3.97 \pm 1.39) \times 10^{-3}$	$(4.25 \pm 1.07) \times 10^{-3}$
$M = 64$	$(9.79 \pm 1.95) \times 10^{-3}$	$(9.82 \pm 1.89) \times 10^{-3}$	$(9.77 \pm 1.90) \times 10^{-3}$
$M = 128$	$(6.11 \pm 1.37) \times 10^{-3}$	$(6.16 \pm 1.30) \times 10^{-3}$	$(6.07 \pm 1.30) \times 10^{-3}$
$M = 512$	$(1.91 \pm 0.49) \times 10^{-3}$	$(1.89 \pm 0.39) \times 10^{-3}$	$(1.93 \pm 0.62) \times 10^{-3}$

Table 11: gPC Order (p) Ablation

Configuration	$\mathcal{E}_{\bar{u}}$	$\mathcal{E}_{u_1}$	$\mathcal{E}_{u_2}$
default (baseline)	$(4.23 \pm 1.03) \times 10^{-3}$	$(3.97 \pm 1.39) \times 10^{-3}$	$(4.25 \pm 1.07) \times 10^{-3}$
p=1	$(3.85 \pm 1.75) \times 10^{-3}$	$(4.30 \pm 1.91) \times 10^{-3}$	$(4.44 \pm 1.68) \times 10^{-3}$
p=2	$(3.50 \pm 1.06) \times 10^{-3}$	$(3.54 \pm 1.03) \times 10^{-3}$	$(3.51 \pm 1.12) \times 10^{-3}$
p=4	$(2.56 \pm 3.43) \times 10^{-2}$	$(2.35 \pm 3.08) \times 10^{-2}$	$(2.55 \pm 3.41) \times 10^{-2}$

Table 12: CP Rank (q) Ablation

Configuration	$\mathcal{E}_{\bar{u}}$	$\mathcal{E}_{u_1}$	$\mathcal{E}_{u_2}$
$q = 4$	$(4.74 \pm 1.84) \times 10^{-3}$	$(4.66 \pm 1.79) \times 10^{-3}$	$(4.79 \pm 1.89) \times 10^{-3}$
$q = 8$	$(2.80 \pm 1.04) \times 10^{-3}$	$(2.73 \pm 1.09) \times 10^{-3}$	$(2.85 \pm 0.93) \times 10^{-3}$
$q = 16$ (*default*)	$(4.23 \pm 1.03) \times 10^{-3}$	$(3.97 \pm 1.39) \times 10^{-3}$	$(4.25 \pm 1.07) \times 10^{-3}$
$q = 32$	$(3.24 \pm 0.91) \times 10^{-3}$	$(3.43 \pm 0.42) \times 10^{-3}$	$(3.15 \pm 0.83) \times 10^{-3}$

G Scaling Study: Stochastic Dimension and CP Rank

To characterize how S^2-PINN scales with stochastic dimensionality R and CP rank q, we sweep $R \in \{3, 5, 8, 10\}$ and $q \in \{8, 16, 32\}$ on the stochastic diffusion benchmark with p=3 fixed. Table 13 shows the full results including wall-clock time and peak GPU memory. The $R = 15$ setting is included as a larger stress test and is evaluated for $q \in \{16, 32\}$.

The table supports three conclusions. First, mean and variance errors remain in the 10^{-3} to 10^{-2} range through the diffusion stress test, including R=15 where the gPC basis has G=816 modes. Second, larger CP rank is not uniformly better on this benchmark: q=8 is competitive for $R \leq 10$, while q=32 improves the reported R=15 run. Third, wall-clock time increases with stochastic dimension, but peak GPU memory remains close to 1 GB, consistent with the low-rank CP coupling avoiding a dense MM_tG tensor. For reference, R=15 with p=3 already yields $G=\binom{18}{3}$=816 stochastic modes, comparable to the number of retained modes in KL-based stochastic modeling studies of heterogeneous media. To probe extreme scaling, we additionally run R=50 (G=23,426): the mean and variance errors remain at the 10^{-2} level, while peak memory grows to about 18 GB, reflecting the polynomial growth of G with R (Appendix I).

Table 13: Scaling sweep on stochastic diffusion: mean/variance relative L^2 error, wall-clock time, and peak GPU memory across stochastic dimension R and CP rank q (p=3, M=256, 20K steps.) *The R=50 run (G=23,426) uses learning rate 10^{-4} and an NVIDIA L40S GPU because of its ≈18 GB peak memory; its time is therefore not directly comparable with the T4 runs.

R	q	Mean rel. L^2	Var. rel. L^2	Time (min)	Mem (GB)
3	8	$(5.88 \pm 1.83) \times 10^{-3}$	$(8.00 \pm 2.37) \times 10^{-3}$	84.6 ± 0.9	0.99
3	16	$(5.81 \pm 0.89) \times 10^{-3}$	$(7.32 \pm 1.38) \times 10^{-3}$	84.9 ± 0.6	0.99
3	32	$(5.37 \pm 0.66) \times 10^{-3}$	$(9.61 \pm 2.98) \times 10^{-3}$	84.3 ± 0.5	0.99
5	8	$(4.50 \pm 0.31) \times 10^{-3}$	$(6.96 \pm 2.23) \times 10^{-3}$	93.6 ± 0.4	0.99
5	16	$(6.43 \pm 1.71) \times 10^{-3}$	$(1.26 \pm 0.45) \times 10^{-2}$	93.6 ± 0.6	0.99
5	32	$(4.58 \pm 0.94) \times 10^{-3}$	$(7.94 \pm 1.29) \times 10^{-3}$	94.0 ± 0.2	0.99
8	8	$(5.86 \pm 0.73) \times 10^{-3}$	$(8.14 \pm 1.46) \times 10^{-3}$	146.1 ± 1.1	0.99
8	16	$(5.81 \pm 2.25) \times 10^{-3}$	$(8.50 \pm 2.17) \times 10^{-3}$	146.7 ± 0.4	0.99
8	32	$(6.41 \pm 1.72) \times 10^{-3}$	$(8.69 \pm 1.02) \times 10^{-3}$	146.1 ± 1.0	1.00
10	8	$(4.27 \pm 0.62) \times 10^{-3}$	$(5.57 \pm 0.98) \times 10^{-3}$	233.6 ± 1.7	0.99
10	16	$(5.49 \pm 2.30) \times 10^{-3}$	$(7.49 \pm 3.44) \times 10^{-3}$	233.8 ± 1.8	0.99
10	32	$(5.16 \pm 1.78) \times 10^{-3}$	$(6.10 \pm 1.81) \times 10^{-3}$	234.5 ± 1.0	0.99
15	16	$(7.30 \pm 1.55) \times 10^{-3}$	$(9.46 \pm 2.84) \times 10^{-3}$	677.3 ± 1.7	1.06
15	32	$(4.18 \pm 0.52) \times 10^{-3}$	$(5.40 \pm 0.39) \times 10^{-3}$	679.5 ± 2.3	1.06
50*	–	$(1.09 \pm 0.14) \times 10^{-2}$	$(1.24 \pm 0.30) \times 10^{-2}$	131.9 ± 0.3	18.0

H Collocation Density Sensitivity

We vary the number of collocation points $N_c \in \{4096, 8192, 16384, 32768, 65536\}$ on the stochastic diffusion benchmark (R=5, p=3, q=16, 20K steps) to characterize sensitivity to spatial sampling density. Table 14 reports the results.

Mean error decreases modestly from 6.34×10^{-3} at N_c=4096 to 4.68×10^{-3} at N_c=65536, a $1.4\times$ improvement over a $16\times$ increase in batch size. Variance error is also lowest at the largest batch, while calibration error is not monotone. We therefore use N_c=32,768 as a practical default rather than claiming that larger collocation batches are uniformly better.

Table 14: Collocation density sweep on stochastic diffusion (R=5, p=3, q=16, 20K steps).

N_c	Mean rel. L^2	Var. rel. L^2	Cal. error
4096	$(6.34 \pm 3.27) \times 10^{-3}$	$(1.30 \pm 1.25) \times 10^{-2}$	$(1.85 \pm 0.32) \times 10^{-2}$
8192	$(5.95 \pm 1.58) \times 10^{-3}$	$(9.70 \pm 3.23) \times 10^{-3}$	$(1.47 \pm 0.53) \times 10^{-2}$
16384	$(5.72 \pm 0.79) \times 10^{-3}$	$(9.54 \pm 1.75) \times 10^{-3}$	$(1.66 \pm 0.72) \times 10^{-2}$
32768	$(6.43 \pm 1.71) \times 10^{-3}$	$(1.26 \pm 0.45) \times 10^{-2}$	$(1.27 \pm 0.35) \times 10^{-2}$
65536	$(4.68 \pm 0.56) \times 10^{-3}$	$(8.49 \pm 5.49) \times 10^{-3}$	$(1.94 \pm 0.52) \times 10^{-2}$

I Computational Cost with Respect to q, M, and R

CP core. For a batch of size B, the forward map is $u_b = \sum_{k=1}^{q} w_k (\mathbf{F}_{\text{space}} A)_{bk} (\mathbf{F}_{\text{time}} B_t)_{bk} (\mathbf{F}_{\text{stoc}} D)_{bk}$ with $A \in \mathbb{R}^{M\times q}$, $B_t \in \mathbb{R}^{M_t\times q}$, and $D \in \mathbb{R}^{G\times q}$. The CP core therefore has exactly $P_{\text{CP}} = q(M + M_t + G + 1)$ parameters, and the three matrix products cost $O(BMq)$, $O(BM_tq)$, and $O(BGq)$, i.e., $T_{\text{CP}} = O\big(Bq(M + M_t + G)\big)$. A dense coupling tensor would require $P_{\text{dense}} = MM_tG$ parameters and $T_{\text{dense}} = O(BMM_tG)$ operations, so the CP coupling is cheaper whenever $q \ll MM_tG/(M + M_t + G)$.

Cost in q. Each additional CP component adds $\Delta_q P_{\text{CP}} = M + M_t + G + 1$ parameters and $\Delta_q T_{\text{CP}} = O\big(B(M + M_t + G)\big)$ operations; for fixed M, M_t, G we have $P_{\text{CP}} = \Theta(q)$ and $T_{\text{CP}} = \Theta(Bq)$. The best rank-$q$ approximation error is non-increasing in q, because rank-$(q-1)$ tensors are contained in the rank-q class. The accuracy of a finitely trained model, however, need not improve monotonically, because optimization is non-convex and subject to a fixed step budget, sampling noise, and regularization. This explains the non-monotone behaviour across $q \in \{8, 16, 32\}$ in Tables 12 and 13.

Cost in M. Each additional Gaussian atom adds q entries to A and three dictionary parameters (two centre coordinates and one width), i.e., $\Delta_M P \approx q + 3$, and costs $\Delta_M T = O\big(B(q + 1)\big)$ for feature evaluation and contraction. For fixed remaining sizes, $P = \Theta(Mq)$ and $T = \Theta(BMq)$. First- and second-order spatial derivatives only multiply the BM term by a constant that depends on the derivative order of the PDE and do not change the linear dependence on M.

Cost in R. The number of total-degree gPC modes is $G(R, p) = \binom{R+p}{p}$; increasing $R \to R + 1$ adds $\Delta_R G = \binom{R+p}{p-1}$ modes, hence $\Delta_R P_{\text{CP}} = q\binom{R+p}{p-1}$ and $\Delta_R T_{\text{CP}} = O\big(Bq\binom{R+p}{p-1}\big)$. For fixed p, $G(R, p) = \Theta(R^p/p!)$, so that

$$P_{\text{CP}} = O\Big(qM + qM_t + q\tfrac{R^p}{p!}\Big), \qquad T_{\text{CP}} = O\Big(Bq\big(M + M_t + \tfrac{R^p}{p!}\big)\Big).$$

Thus, for a fixed degree p, the CP core grows polynomially (of degree p) in R rather than exponentially; the curse of dimensionality is mitigated but not eliminated.

gPC basis evaluation. Evaluating all G tensorized modes costs $T_{\text{gPC,impl}} = O(BRp + BRG)$, where BRp accounts for the univariate polynomials of each random variable and BRG for assembling the tensorized modes. Since every total-degree multi-index with $|\boldsymbol{\alpha}| \le p$ has at most p non-zero entries, a sparse multi-index evaluation reduces this to $T_{\text{gPC,sparse}} = O(BRp + BpG)$.

Projected loss and total training cost. The projected residual adds $T_{\mathrm{proj}} = O(BG)$ for the residual–basis products and reductions. Writing C_{PDE} for the constant factor of the PDE residual and its automatic differentiation, one training step costs

$$T_{\mathrm{step}} = O\Big(C_{\mathrm{PDE}}\, B\big[M + M_t + Rp + RG + q(M + M_t + G)\big] + BG\Big),$$

where RG can be replaced by pG with sparse gPC evaluation, and S training steps cost $T_{\mathrm{train}} = S\,T_{\mathrm{step}}$. The activation memory is approximately $O\big(B(M + M_t + G + q)\big)$, and the memory for parameters and optimizer states is $O\big(q(M + M_t + G) + M\big)$.

Accuracy versus nominal capacity. The retained index set Λ (determined by R and p) and the CP rank q play different roles: $G = \binom{R+p}{p}$ is the number of retained gPC modes, whereas q is the separation rank. For an orthonormal expansion $u = \sum_{\boldsymbol{\alpha}} u_{\boldsymbol{\alpha}} \Psi_{\boldsymbol{\alpha}}$, the best-approximation error of Λ is $\|u - \mathsf{P}_\Lambda u\|_{L^2}^2 = \sum_{\boldsymbol{\alpha} \notin \Lambda} |u_{\boldsymbol{\alpha}}|^2$. Enlarging R or p strictly reduces this truncation error only when the newly included coefficients carry non-negligible stochastic energy, and a larger space also increases estimation and optimization difficulty under a fixed sampling and training budget. This is visible in Tables 11 and 13: q=8 is competitive for $R \le 10$, whereas q=32 helps at R=15, and p=2 outperforms p=4 in the trained model. Larger R, p, and q increase nominal capacity, but by themselves do not guarantee better finite-budget accuracy.

J Stochastic Navier–Stokes Benchmark

We consider the 2D incompressible Navier–Stokes equations in vorticity form,

$$\partial_t \omega + \mathbf{u} \cdot \nabla \omega = \nu(\boldsymbol{\xi}) \Delta \omega + f, \qquad \nabla \cdot \mathbf{u} = 0, \qquad \omega = \nabla \times \mathbf{u},$$

on $\Omega = [0, 2\pi]^2$, $t \in [0, 5]$, with $\boldsymbol{\xi} \sim \mathcal{U}([-1, 1]^5)$, where $\mathbf{u} = (u_1, u_2)$ is the velocity, ω the vorticity, and the pressure has been eliminated. The five random variables control the two phase shifts, the viscosity, and the amplitudes of two vortex modes:

$$X = x + 0.25\xi_1, \quad Y = y + 0.25\xi_2, \quad \nu(\boldsymbol{\xi}) = 0.05(1 + 0.2\xi_3) \in [0.04, 0.06],$$

$$A(\boldsymbol{\xi}) = 1 + 0.1\xi_4 \in [0.9, 1.1], \quad B(\boldsymbol{\xi}) = 0.5(1 + 0.2\xi_5) \in [0.4, 0.6].$$

With $a = Ae^{-2\nu t}$ and $b = Be^{-5\nu t}$, the stream function is $\psi = a \sin X \sin Y + b \sin(2X) \sin Y$, and the exact velocity and vorticity are

$$u_1 = \partial_Y \psi = a \sin X \cos Y + b \sin(2X) \cos Y, \quad u_2 = -\partial_X \psi = -a \cos X \sin Y - 2b \cos(2X) \sin Y,$$

$$\omega = \partial_X u_2 - \partial_Y u_1 = 2a \sin X \sin Y + 5b \sin(2X) \sin Y.$$

This construction satisfies incompressibility exactly; the linear time-derivative and diffusion terms cancel, while the two modes produce a non-zero nonlinear interaction. The resulting analytical forcing is $f(t, x, y, \boldsymbol{\xi}) = -6ab \sin^3 X \sin Y \cos Y$. Results are reported in Tables 15 and 16. Neural Chaos is again a supervised oracle.

Table 15: Stochastic Navier–Stokes: relative L^2 errors of the velocity mean and variance.

Method	u_1 mean	u_2 mean	u_1 variance	u_2 variance
S^2-PINN (proj)	$\mathbf{(2.05 \pm 0.34) \times 10^{-3}}$	$\mathbf{(2.40 \pm 0.21) \times 10^{-3}}$	$\mathbf{(3.13 \pm 0.49) \times 10^{-2}}$	$\mathbf{(2.58 \pm 0.71) \times 10^{-2}}$
S^2-PINN (no-proj)	$(1.09 \pm 0.36) \times 10^{-2}$	$(1.15 \pm 0.27) \times 10^{-2}$	$(2.10 \pm 0.10) \times 10^{-1}$	$(1.27 \pm 0.03) \times 10^{-1}$
Vanilla PINN	$(3.44 \pm 0.43) \times 10^{-2}$	$(3.17 \pm 1.22) \times 10^{-2}$	$(2.15 \pm 0.21) \times 10^{-1}$	$(1.89 \pm 0.17) \times 10^{-1}$
SPINN+Z	$(1.70 \pm 0.62) \times 10^{-2}$	$(1.79 \pm 0.34) \times 10^{-2}$	$(2.63 \pm 0.81) \times 10^{-1}$	$(2.05 \pm 0.82) \times 10^{-1}$
Functional CP-PINN	$(3.48 \pm 0.75) \times 10^{-2}$	$(2.83 \pm 0.74) \times 10^{-2}$	$(1.39 \pm 0.39) \times 10^{-1}$	$(9.81 \pm 3.49) \times 10^{-2}$
Neural Chaos (supervised oracle)	$(1.98 \pm 1.10) \times 10^{-2}$	$(1.93 \pm 0.64) \times 10^{-2}$	$(7.91 \pm 1.09) \times 10^{-2}$	$(1.40 \pm 0.38) \times 10^{-1}$

K Additional Baseline Results

Table 17 complements Table 3 with the calibration of PI-DeepONet (all metrics) and of SPINN+Z, functional CP-PINN, and Neural Chaos (calibration error only), and Table 18 reports training time and peak GPU memory.

Table 16: Stochastic Navier–Stokes: velocity calibration. Coverage values are fractions; Cov90 and Cov95 are better when closer to 0.900 and 0.950, and the calibration error is better when smaller.

Method	Component	Cov90	Cov95	Cal. Err.
S^2-PINN (proj)	u_1	$\mathbf{0.886 \pm 0.002}$	$\mathbf{0.938 \pm 0.002}$	$\mathbf{0.011 \pm 0.003}$
	u_2	$\mathbf{0.892 \pm 0.001}$	$\mathbf{0.944 \pm 0.001}$	$\mathbf{0.007 \pm 0.001}$
S^2-PINN (no-proj)	u_1	0.771 ± 0.026	0.837 ± 0.024	0.119 ± 0.025
	u_2	0.854 ± 0.016	0.909 ± 0.015	0.041 ± 0.015
Vanilla PINN	u_1	0.753 ± 0.015	0.811 ± 0.015	0.128 ± 0.014
	u_2	0.793 ± 0.011	0.852 ± 0.011	0.095 ± 0.011
SPINN+Z	u_1	0.754 ± 0.008	0.810 ± 0.008	0.129 ± 0.007
	u_2	0.803 ± 0.020	0.857 ± 0.018	0.085 ± 0.018
Functional CP-PINN	u_1	0.828 ± 0.023	0.881 ± 0.021	0.061 ± 0.022
	u_2	0.841 ± 0.015	0.894 ± 0.013	0.051 ± 0.014
Neural Chaos (supervised oracle)	u_1	$\mathbf{0.886 \pm 0.013}$	0.935 ± 0.011	$\mathbf{0.012 \pm 0.009}$
	u_2	0.887 ± 0.003	0.937 ± 0.003	0.010 ± 0.002

Table 17: Calibration of additional baselines (same protocol as Table 3). ‡Supervised oracle with one shared checkpoint for diffusion, Allen–Cahn, and Burgers; see Table 1.

Method / PDE	Cov@50%	Cov@80%	Cov@90%	Cov@95%	Sharpness_{90}	Cal. Err.
PI-DeepONet						
Diffusion	0.414 ± 0.048	0.678 ± 0.064	0.770 ± 0.066	0.819 ± 0.062	0.197 ± 0.002	0.117 ± 0.060
Allen–Cahn	0.413 ± 0.046	0.676 ± 0.062	0.767 ± 0.063	0.815 ± 0.060	0.199 ± 0.001	0.120 ± 0.058
Burgers	0.461 ± 0.024	0.741 ± 0.031	0.834 ± 0.030	0.878 ± 0.028	0.202 ± 0.005	0.059 ± 0.028
Darcy	0.448 ± 0.053	0.723 ± 0.060	0.827 ± 0.052	0.887 ± 0.044	0.095 ± 0.001	0.067 ± 0.050
SPINN+Z (calibration error only)						
Diffusion	–	–	–	–	–	0.013 ± 0.003
Allen–Cahn	–	–	–	–	–	0.011 ± 0.002
Burgers	–	–	–	–	–	0.013 ± 0.001
Darcy	–	–	–	–	–	0.008 ± 0.001
Functional CP-PINN (calibration error only)						
Diffusion	–	–	–	–	–	0.153 ± 0.054
Allen–Cahn	–	–	–	–	–	0.173 ± 0.064
Burgers	–	–	–	–	–	0.159 ± 0.038
Darcy	–	–	–	–	–	0.050 ± 0.025
Neural Chaos‡ (calibration error only)						
Diffusion	–	–	–	–	–	0.054 ± 0.006
Allen–Cahn	–	–	–	–	–	–
Burgers	–	–	–	–	–	–
Darcy	–	–	–	–	–	0.093 ± 0.070

L Visualizations of Stochastic Fields

To visually compare spatial stochastic structure across PDE families, we present moment heatmaps (mean/variance) at t=1.0 for diffusion, Allen–Cahn, Burgers, and the two non-manufactured benchmarks (Poisson MC and Darcy MC). Each panel includes both 2D fields and 3D surfaces for reference, model prediction, and absolute error.

Table 18: Training time and peak GPU memory on stochastic diffusion (R=5).

	S^2-PINN	Vanilla PINN	PI-DeepONet	SPINN+Z	Func. CP-PINN	Neural Chaos
Time (min)	≈90	≈80	≈120	≈100	≈100	≈120
Peak memory (GB)	1.0	4.0	5.0	0.3	0.3	1.5

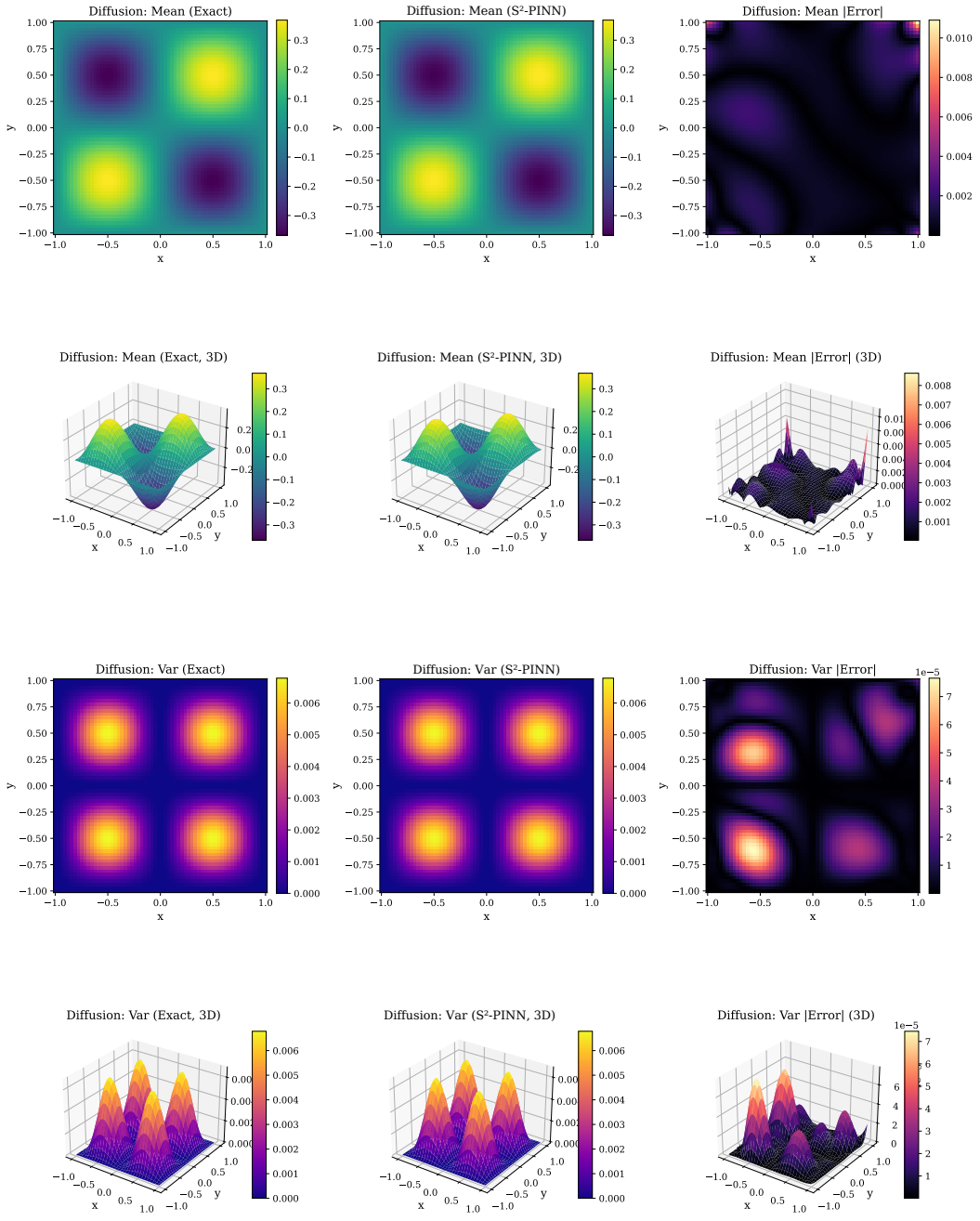

Figure 4: Diffusion heatmaps (2D and 3D) at t=1.0: reference mean/variance, model-predicted mean/variance, and absolute errors.

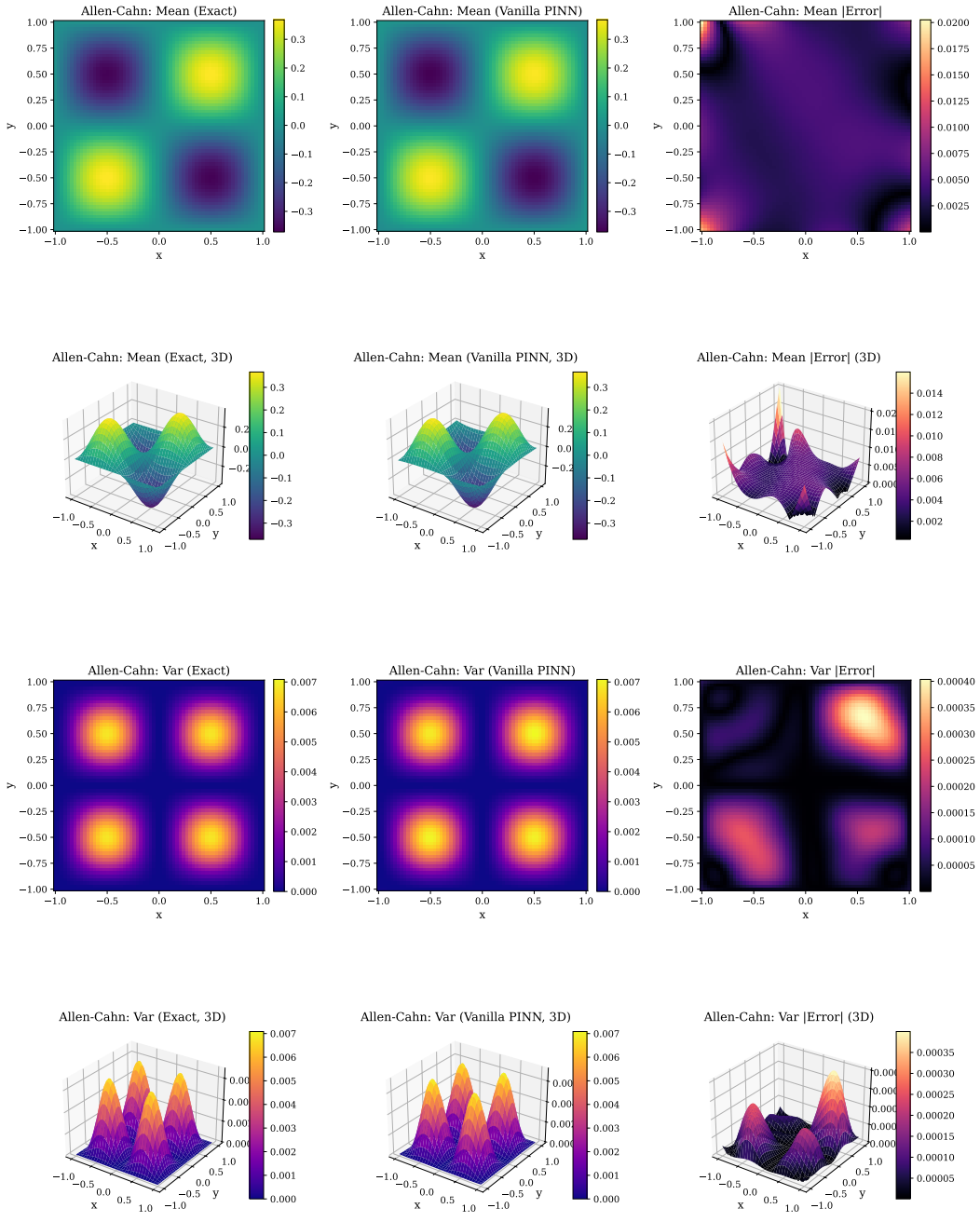

Figure 5: Allen–Cahn heatmaps (2D and 3D) at t=1.0: reference mean/variance, model-predicted mean/variance, and absolute errors.

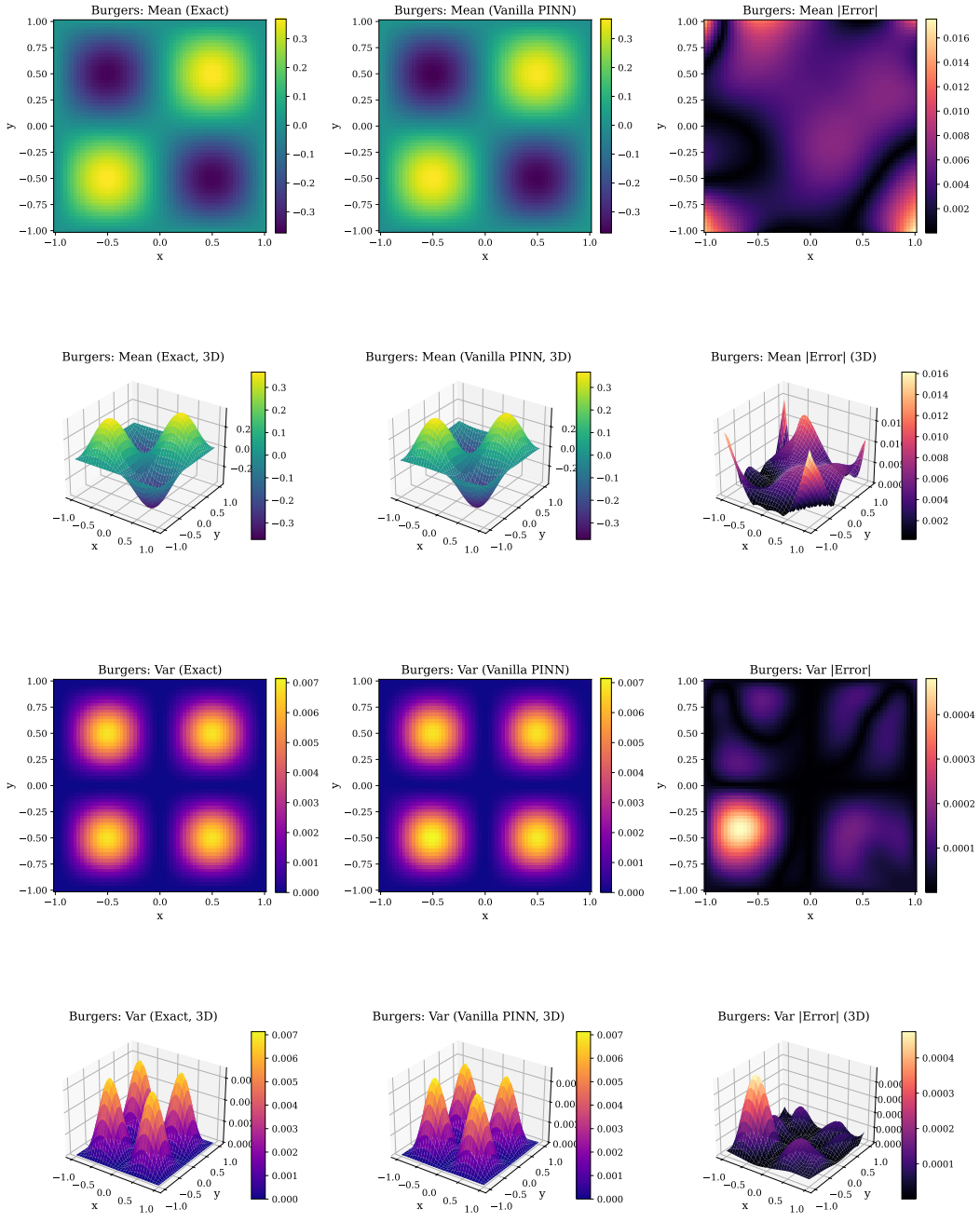

Figure 6: Burgers heatmaps (2D and 3D) at t=1.0: reference mean/variance, model-predicted mean/variance, and absolute errors.

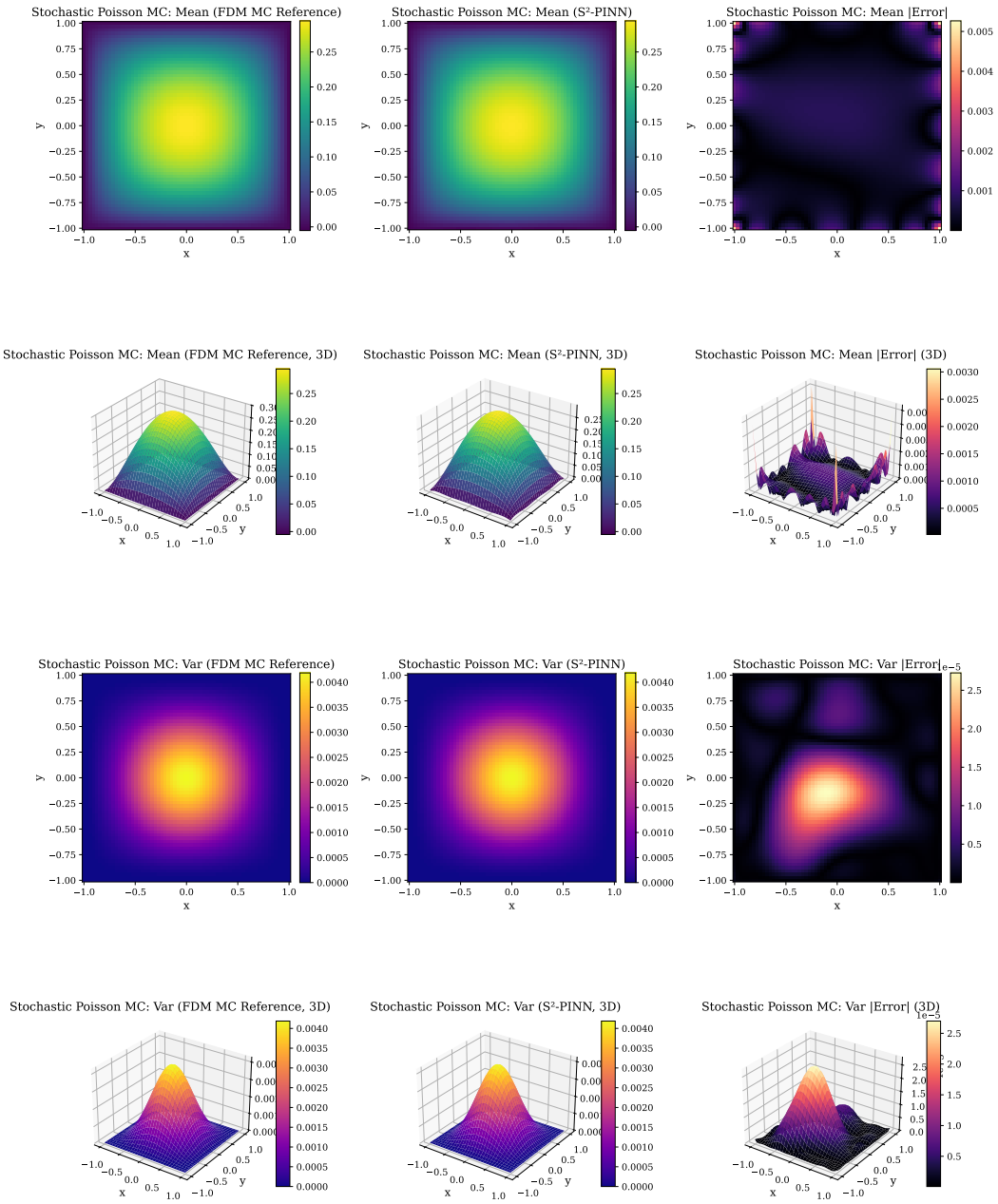

Figure 7: Non-manufactured Poisson heatmaps (2D and 3D) at t=1.0: FDM Monte Carlo reference mean/variance (10,000 samples), S^2-PINN prediction, and absolute errors.

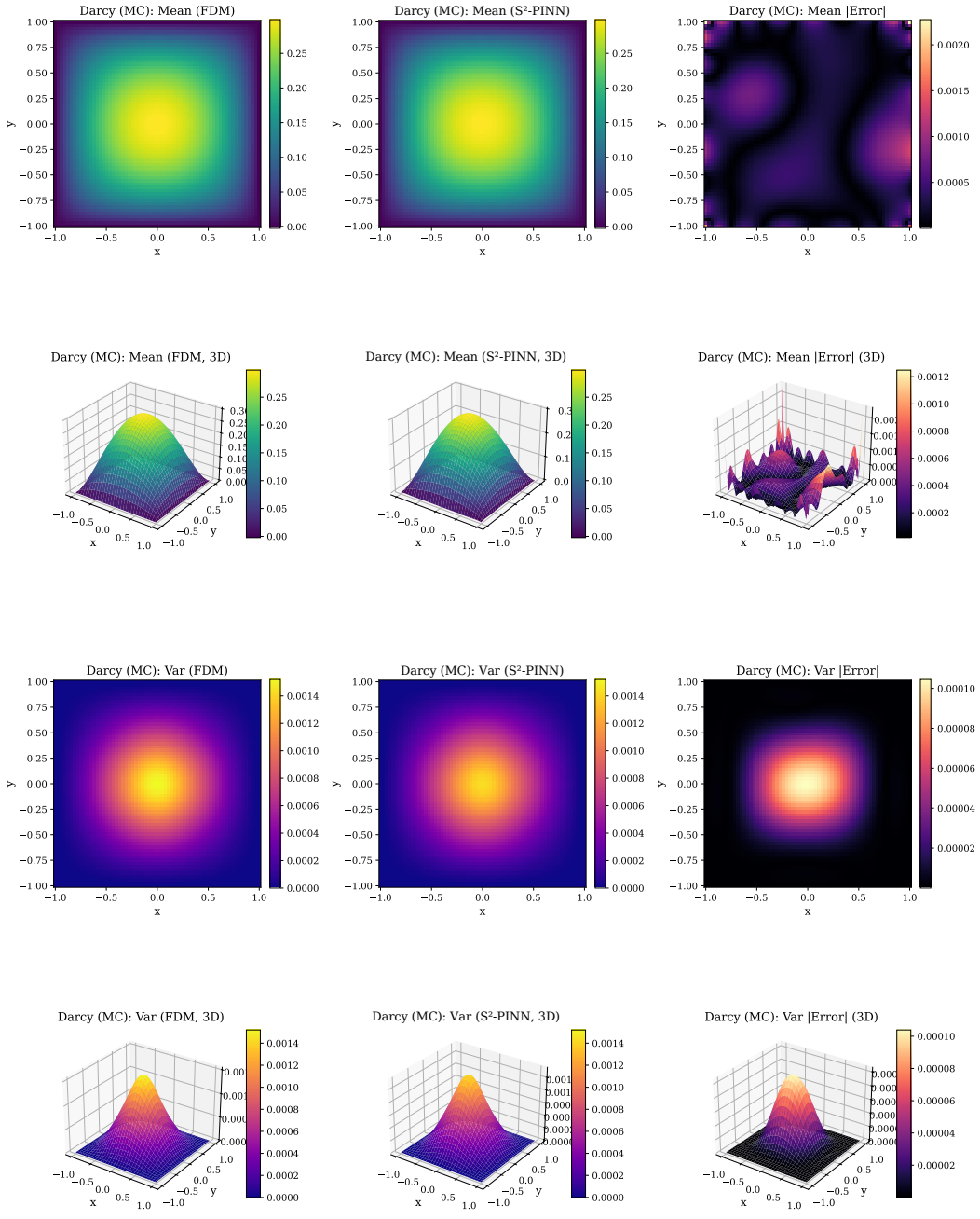

Figure 8: Non-manufactured Darcy heatmaps (2D and 3D) at t=1.0: FDM Monte Carlo reference mean/variance (10,000 samples), S^2-PINN prediction, and absolute errors.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes], [No], or [N/A].
- [N/A] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for [N/A]).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will also be asked to include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While [Yes] is generally preferable to [No], it is perfectly acceptable to answer [No] provided a proper justification is given (e.g., error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering [No] or [N/A] is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction accurately describe the proposed contributions about theoretical and experimental results.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

Justification: We discuss limitations of our work in Section 5

Guidelines:

- The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: We provide our proof for stability and consistency in Appendix B

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Yes, we include all experimental details necessary to reproduce. Pseudo code for the algorithm is given in Algorithm 1 and default parameters are listed in Appendix E.

   Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: You can visit this anonymous link to get the code.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).

- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Default parameters are listed in Appendix E.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: All the results are reported as an average of 5 random seeds (42,123,456,789.1024) along with mean and standard deviation.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: All the experiments are run on NVIDIA T4 and it is reported in Section E.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.

- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Yes, the research conducted complies with the NeurIPS ethical guidelines in all respects.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes],

    Justification: They are reported in Section 5.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper doesn't include any pre-trained models or datasets.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Yes, all assets have been cited appropriately.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: This research does not release any new assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: This research does not involve any human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [No]

Justification: This research presented does not involve LLMs as any important, original, or non-standard components.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.